Aalto University
School of Engineering
Degree Programme in Energy and HVAC-Technology

Pyry Jantunen

Verification and Validation of a Computational Fluid Dynamics Code

Master's Thesis
Espoo, October 13, 2017

Supervisor: Professor Jukka Tuhkuri
Advisors: Professor Emeritus Timo Siikonen
Juho Ilkko M.Sc. (Tech)

Aalto University
School of Engineering
Degree Programme in Energy and HVAC-Technology

ABSTRACT OF
MASTER'S THESIS

Author:	Pyry Jantunen		
Title:	Verification and Validation of a Computational Fluid Dynamics Code		
Date:	October 13, 2017	**Pages:**	vii + 65
Major:	Applied Thermodynamics	**Code:**	Ene-39
Supervisor:	Professor Jukka Tuhkuri		
Advisors:	Professor Emeritus Timo Siikonen Juho Ilkko M.Sc. (Tech)		

Computational fluid dynamics as an applied science aims to describe the governing equations of a fluid flow in a form implementable on a computer. To reach such a description, a multitude of simplifications and approximations are needed, especially in a case of a turbulent flow. An accurate solution of the smallest turbulent eddies would require a number of numerical operations infeasible in practice. Replacing the accurate governing equations with turbulence models, which fade out the smallest eddies, allows for an application of fluid dynamics in engineering problems. However, such modeling naturally deviates the computed results more or less from real flows. Also, a numerical solution given by a computer does not equal the analytical solution of the governing equations. This is true whether the equations are laws of nature or mere models. Thus, the credibility of a flow simulation is never self-evident, and the evaluation of both computational fluid dynamics codes and individual simulations alike is a necessity sometimes overlooked.

The main objective of this master's thesis is the numerical analysis and verificaton of the code FINFLO. The method of the analysis is a comparison with other codes verified in a recent publication. A brief validation study, a comparison of computed and experimental data, is also included. The verification and validation are preceded by a theoretical background overview. This consists of the turbulence models employed and the solution of continuous equations by numerical methods suitable for fluid dynamics.

The verification study revealed some unexpected details about the turbulence models and the computer round-off error. After some fixes, the FINFLO code was observed to behave like the other verified codes. This increases the credibility of all the codes reviewed. The validation study provided expected results, demonstrating the limits of the turbulence models used.

Keywords:	FINFLO, computational fluid dynamics, CFD, verification, validation, turbulence, Spalart-Allmaras, SST $k-\omega$, RANS
Language:	English

Aalto-yliopisto
Insinööritieteiden korkeakoulu
Energia- ja LVI-tekniikan koulutusohjelma

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä:	Pyry Jantunen		
Työn nimi:	Virtausratkaisijan Verifiointi ja Validointi		
Päiväys:	13. lokakuuta 2017	**Sivumäärä:**	vii + 65
Pääaine:	Sovellettu Termodynamiikka	**Koodi:**	Ene-39
Valvoja:	Professori Jukka Tuhkuri		
Ohjaajat:	Professori Emeritus Timo Siikonen Diplomi-insinööri Juho Ilkko		

Virtauslaskenta soveltavana tieteenalana pyrkii saattamaan virtausta kuvaavat luonnonlait tietokoneelle ohjelmoitavaan muotoon. Tämä vaatii joukon yksinkertaistuksia ja oletuksia, eritoten turbulenttisessa virtauksessa, missä pienimpien pyörteitten huomioon ottaminen vaatii kohtuuttomasti laskentatyötä. Vähemmällä päästään soveltamalla tarkkojen luonnonlakien sijaan pyörteet häivyttäviä turbulenssimalleja, mikä mahdollistaa käytännön virtausongelmien ratkaisun säädyllisessä ajassa nykyisillä tietokoneilla, mutta osaltaan etäännyttää laskennan tulosta todellisesta virtauksesta. Valitettavan vähälle huomiolle jää usein myös se, ettei tietokoneen antama numeerinen ratkaisu ole yhtä kuin käytettyjen yhtälöiden analyyttinen ratkaisu, olivat yhtälöt sitten malleja tai luonnonlakeja. Virtauslaskennan tarkkuus ei täten ole mikään itsestäänselvyys, vaan vaatii sekä ohjelmisto- että ongelmakohtaista arviointia.

Tämän diplomityön ensisijainen tarkoitus on virtausratkaisija FINFLOn numeerinen tarkastelu, verifiointi, vertaamalla tätä hiljattain julkaistussa artikkelissa verifioituihin muihin ratkaisijoihin. Työ sisältää myös lyhyen validointitutkimuksen eli laskennallisten tulosten vertailun kokeellisiin. Verifiointia ja validointia pohjustetaan käymällä läpi käytetyn turbulenssimallinnuksen perusta sekä jatkuvien yhtälöiden likimääräinen ratkaisu numeerisin menetelmin.

Verifiointitutkimus paljasti joitain odottamattomia seikkoja eri turbulenssimalleista ja tietokoneen pyöristysvirheestä. Korjausten jälkeen FINFLOn todettiin lopulta käyttäytyvän samoin kuin vertailuohjelmistojen, kasvattaen näiden kaikkien luotettavuutta. Validointi sujui yllätyksettömämmin, osoittaen käytettyjen turbulenssimallien soveltuvuuden rajat.

Asiasanat:	FINFLO, virtauslaskenta, virtausmekaniikka, verifiointi, validointi, turbulenssi, Spalart-Allmaras, SST $k-\omega$, RANS
Kieli:	Englanti

Acknowledgements

I wish to thank my supervisor Professor Jukka Tuhkuri and advisors Professor Emeritus Timo Siikonen and Juho Ilkko for their guidance and the opportunity to work on this thesis. I would also like to thank Senior Lecturer Kari Santaoja and Finflo employee Esa Salminen, whose help was invaluable in completing this.

A number of free software was used in making this thesis. The Latex files of which the written part consists were edited with TeXstudio, while the figures were drawn with Python and Matplotlib. The workstation on which all this was done runs a Fedora Linux distribution.

The work was funded by the Foundation for Aalto University Science and Technology.

In memory of my grandfather, Onni Jantunen, who was looking forward to seeing me graduate but passed away a little too early,

Espoo, October 13, 2017

Pyry Jantunen

Nomenclature

a	speed of sound
B	turbulent wall-law intercept constant
C_D	drag coefficient
C_{Dp}	pressure drag coefficient
C_{Dv}	viscous drag coefficient
C_{fx}	x-wise wall friction coefficient
C_L	lift coefficient
C_M	pitching moment coefficient
C_p	pressure coefficient
c_p	specific heat capacity in constant pressure
c_v	specific heat capacity in constant volume
E	total energy; estimated fractional error
e	specific internal energy
F	flux
h	specific enthalpy
I	turbulent intensity
k	turbulent kinetic energy
L_l	difference of outgoing and incoming fluxes in l-direction
Ma	Mach number
N	number of nodes in a mesh
p	pressure; order of error
Pr	Prandtl number
Pr_t	turbulent Prandtl number
q_j	heat flux component
Re	Reynolds number
S	surface area
T	temperature; a transported quantity to be discretized
$\boldsymbol{T}$	vector of conservative variables
t	time
U	fluid velocity
u_τ	wall friction velocity
u^+	dimensionless velocity parallel to wall
V	volume
W	magnitude of the vorticity
$\boldsymbol{W}$	vector of characteristic variables
y^+	dimensionless wall normal coordinate
$u,\quad v,\quad w$	velocity components
$x,\quad y,\quad z$	orthogonal coordinates

Greek letters

α	angle of attack; a generic diffusion coefficient
γ	heat capacity ratio

Δ	indicator of a change
δ_{ij}	Kronecker's delta function
ϵ	turbulent dissipation
κ	von Kármán constant; MUSCL scheme parameter
μ	dynamic viscosity
μ_t	dynamic eddy/turbulent viscosity
ρ	density
σ_k	Schmidt number for turbulent kinetic energy
$\sigma_{\tilde{\nu}}$	Schmidt number for $\tilde{\nu}$ turbulence variable
σ_ω	Schmidt number for specific turbulent dissipation
σ_{ij}	viscous stress tensor
τ_{ij}	turbulent/Reynolds stress tensor
τ_w	wall shear stress
ν	kinematic viscosity
ν_t	kinematic eddy/turbulent viscosity
$\tilde{\nu}$	Spalart-Allmaras turbulence variable
ω	specific turbulent dissipation

Subscripts

$i, \quad j, \quad k$	component in spatial direction
l	index of a discrete point in space

Superscripts

n	index of a discrete point in time
$'$	turbulent fluctuation
$''$	mass-weighted turbulent fluctuation

Contents

Nomenclature **v**

1 Introduction **1**

2 Reynolds-averaged Navier-Stokes equations **2**
2.1 Navier-Stokes equations . . . 3
2.2 Reynolds-averaging of the Navier-Stokes equations . . . 4
2.3 Favre-averaging of the Navier-Stokes equations . . . 6
2.4 Boussinesq approximation and Reynolds analogy . . . 7
2.5 Turbulent kinetic energy in two-equation models . . . 8
2.6 Turbulent pressure work hypothesis . . . 11
2.7 Internal and kinetic energies . . . 13
2.8 Simplified equations . . . 14
2.9 Turbulent boundary layer . . . 15

3 Verification and validation **18**
3.1 Discretization error . . . 19
3.2 Richardson extrapolation . . . 22
3.3 Computer round-off error . . . 23
3.4 Modeling error . . . 23
3.5 In this work . . . 24
3.6 In philosophy of science . . . 25

4 Flow solution **27**
4.1 FINFLO . . . 27
4.2 Spalart-Allmaras turbulence model . . . 30
4.3 Shear stress transport $k-\omega$ turbulence model . . . 33

5 Benchmark flow **37**
5.1 Grid . . . 38
5.2 Boundary conditions . . . 40
5.3 Grid convergence . . . 41
5.4 Skin friction and pressure . . . 47
5.5 Convergence histories . . . 52
5.6 Flow profiles . . . 53
5.7 Computing time . . . 55
5.8 Angle of attack . . . 56

6 Conclusions **59**

Appendix: Computed coefficients **65**

Chapter 1

Introduction

This research work is made for a master's thesis at the Aalto University School of Engineering. The bulk of this work consists of verifying the Computational Fluid Dynamics (CFD) code FINFLO in problem sizes unprecedented for this code, following the example of Diskin et al. (2016), and comparing the results they obtained from different CFD codes. The code called CFL3D (NASA, 2017a) provides the primary benchmark, as it is close to FINLFO in the approach to model and solve viscous flow equations.

Verification in CFD means purely numerical evaluation (Roache, 1998), but a brief validation study, a comparison with experimental data, is also included in this work. Thus, the sophistically Latin title of this paper could be simply seen as "Testing the accuracy of a CFD code". Nonetheless, verification and validation have established meanings in the field and are more descriptive to those familiar with them.

FINFLO, a commercially used code, has certainly been verified and validated before, e.g. Guillaume et al. (2012) and Laine et al. (1992). Unfortunately, the depth of verification is limited by the computational power available, and more powerful computers can always bring a better understanding of the numerical behavior of the code. The reason to continue validation is more philosophical; a successful validation of a flow case should not be extrapolated to all flows, lest we accept indubitable inductive logic (Popper, 1959). Validation then expands the range of problems where the code is known to work, but can never cover all possible problems. To recapitulate, code verification and validation should not be seen as final proofs of correctness but as steps in confidence building, a task that will never be truly finished.

In this thesis, a large portion of the written part consists of a theoretical background, here chosen to concentrate on turbulence modeling. It helps the reader to understand the basic properties of viscous fluid flow. The next chapter begins with a summary of the Navier-Stokes equations and strives to explain the physics behind time-averaged linear turbulence modeling. A simple hypothesis on the subject is also introduced. The third chapter, Verification and Validation, discusses the terminology, detailed goals, and methods of the work, being more topical to the ultimate subject but does also include some of the author's own thoughts in the end. The fourth chapter, Flow solution, describes the equipment, code, and turbulence models employed. Finally, the computed results are presented and reviewed along with the benchmark simulations and measurements.

The information concerning FINFLO was collected by reading the manual, talking to the company employees, and digging through source code. Consequently, it is poorly citated but should be accurate enough.

Chapter 2

Reynolds-averaged Navier-Stokes equations

The Navier-Stokes equations apparently do apply in the case of a turbulent flow, as the scale of even the smallest eddies is well above the molecular scale and, therefore, the assumption of continuum holds. Unfortunately, Direct Numerical Simulation (DNS) of turbulence, simply solving the Navier-Stokes equations in a turbulent flow, is currently not feasible in practice (White, 2006). In fact, the numbers presented by White suggest that DNS of turbulence in a simple three-dimensional flow over a flat plate of $2.88\,\mathrm{m}^2$ would require computing power over a thousandfold of what was within reach when the book was edited in 2006. While this is a very rough approximation, it is fairly safe to say that the need to model turbulence in more complex engineering applications will continue for the foreseeable future.

Reynolds-averaging is a method for fading out the unpredictable turbulent fluctuations in the mathematical description of a flow field. Applied to Navier-Stokes equations it yields conservation equations for time-averaged mean quantities, which act laminar like and can be solved as such, with a reasonable number of numerical operations. These are the Reynolds-averaged Navier-Stokes (RANS) equations, where the effects of turbulent eddies are reduced into terms called turbulent stresses and fluxes. Thus, it is assumed that a turbulent flow can be sufficiently represented by its mean motion and some added terms, as originally proposed by Reynolds (1894). These terms cannot be directly solved, but need to be approximated by a turbulence model.

In other words, RANS does not consider turbulent eddies as actual motion, but only as increased mixing. This works well when the scale of the eddies is well below what could be considered significant in the scale of flow problem being solved. However, as the scale of eddies grows, so does the inaccuracy of this assumption. This problem does not occur in Large Eddy Simulation (LES), which is an approach to approximate turbulence different from RANS. As the name suggests, LES fully simulates the larger eddies but is less accurate with the smaller ones. The two approaches are combined in Detached Eddy Simulation (DES), where boundary layers, in which the scale of turbulence is small, are simulated with RANS, while the rest of computational domain uses LES (Spalart et al., 1997; Menter et al., 2003).

Only RANS models are employed in this thesis due to time constraints. RANS can be applied two-dimensionally and has a benefit of producing time independent, averaged results with steady turbulent flows, even though turbulence is always truly time dependent and three-dimensional. It also smooths out solutions of unsteady flows, making them easier to postprocess. This and the more lenient mesh density requirements greatly reduce the computation time compared to LES (Spalart et al., 1997), although the mesh requirement is meaningless in this work, as the meshes used in evaluating the discretization error of the code used are unnecessarily dense from a modeling view.

2.1 Navier-Stokes equations

Navier-Stokes equations describe the conservation of mass, momentum and energy for an infinitesimally small control volume of fluid. Using Einsteinian notation for velocity components ($u_1 = u, \quad u_2 = v, \quad u_3 = w$) and spatial directions ($x_1 = x, \quad x_2 = y, \quad x_3 = z$), they can be written

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_j)}{\partial x_j} = 0 \tag{2.1}$$

$$\frac{\partial (\rho u_i)}{\partial t} + \frac{\partial (\rho u_i u_j)}{\partial x_j} = \frac{\partial \sigma_{ij}}{\partial x_j} - \frac{\partial p}{\partial x_i} \tag{2.2}$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_j}[u_j(E+p)] = \frac{\partial}{\partial x_j}(u_i \sigma_{ij} - q_j) \tag{2.3}$$

where ρ is the density, t the time, p the pressure, σ_{ij} the viscous stress tensor, q_j a heat flux component, and the total energy E

$$E = \rho e + \frac{1}{2}\rho u_i u_i \tag{2.4}$$

is given as the sum of density times the specific internal energy e and the kinetic energy $\frac{1}{2}\rho u_i u_i$. Effects of gravity are deemed negligible in the present application of an aerodynamic flow, shortening the equations. The viscous stress tensor is defined

$$\sigma_{ij} = \mu \left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij} \right) \tag{2.5}$$

Here δ_{ij} is the Kronecker's delta function, which is equal to 1 when $i = j$ and otherwise 0. The tensor is obtained using the Stokes' hypothesis, where we assume that the bulk viscosity λ is the appropriate fraction of dynamic viscosity $\lambda = -(2/3)\mu$, in order to equalize mechanical and thermodynamic pressures. The hypothesis has faced some controversy, (White, 2006), but will not be discussed here. Sufficient to say, the effects of the bulk viscosity compared to dynamic the viscosity are notable only in special cases, when the Reynolds number is generally also high and the viscous stress in total is diminished by convection and pressure. The hypothesis is used in FINFLO as well as in all known CFD codes.

In the energy equation the heat flux is calculated from

$$q_j = -\frac{\mu c_p}{Pr}\frac{\partial T}{\partial x_j} \tag{2.6}$$

where c_p is the specific heat capacity in constant pressure, Pr the Prandtl number, and T the temperature. Prandlt number is a fluid property and denotes the ratio of momentum and heat diffusion coefficients in a fluid, so that $Pr = \mu c_p / k$, where k is the thermal conductivity of the fluid. Writing the heat flux in this way allows us to draw a clear analogy between laminar and turbulent fluxes later on.

2.2 Reynolds-averaging of the Navier-Stokes equations

Reynolds-averaging begins by splitting the calculated quantities into time averaged and fluctuating parts as

$$f = \overline{f} + f' \tag{2.7}$$

where f is the time accurate quantity, $\overline{f}$ its average over a period of time and f' its turbulent fluctuation, illustrated in Fig. 2.1. Density fluctuations are assumed insignificant, simplifying the averaging, but also making it inaccurate for compressible flows. The same assumption is applied to fluid properties, which in reality are functions of a turbulent temperature field. Density and fluid property averaging are handled in immaculate detail by Favre-averaging, briefly discussed in the next section.

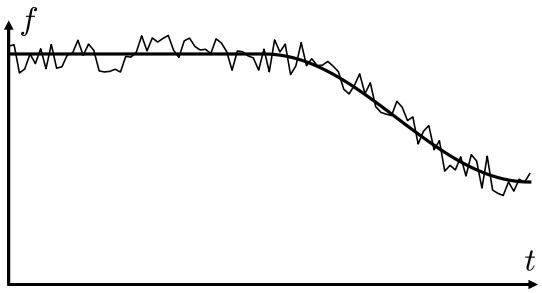

Figure 2.1 Quantity f in a turbulent flow that begins steady and turns unsteady. The thin line shows time accurate quantity and the thick line its average.

The following rules are applied to average the quantities f and g (White, 2006):

$$\overline{f'} = 0 \qquad \overline{fg} = \overline{f}\,\overline{g} + \overline{f'g'} \qquad \overline{f+g} = \overline{f} + \overline{g} \qquad \overline{f\overline{g}} = \overline{f}\,\overline{g} \qquad \overline{\frac{\partial f}{\partial s}} = \frac{\partial \overline{f}}{\partial s} \tag{2.8}$$

The velocity components, viscous stress tensor, pressure, internal energy, and temperature are all split according to Eq. (2.7). Time averaging terms that only have one fluctuating quantity reduces them into the averaged part, e.g. $\overline{\rho u_i} = \overline{\rho(\bar{u}_i + u_i')} = \rho\bar{u}_i$. However, terms that contain two or more fluctuating quantities produce new terms, the turbulent stresses and fluxes mentioned earlier. Applying the second rule in Eqs. (2.8) to the averaged momentum convection gives

$$\overline{\rho u_i u_j} = \rho\bar{u}_i\bar{u}_j + \rho\overline{u_i'u_j'}$$

where we cannot assume that the product of two fluctuations would reduce to zero when averaged, but must include it as an unknown tensor in the momentum equation. This is often called the Reynolds stress tensor, here defined as $-\rho\overline{u_i'u_j'}$. The definition varies with the method of averaging, and will be different once the density fluctuations are accounted for.

The averaged energy equation is a bit more laborious to derive. First, internal energy convection and pressure work are combined

$$u_j(E+p) = u_j(\rho e + \frac{1}{2}\rho u_i u_i + p) = u_j(\rho h + \frac{1}{2}\rho u_i u_i)$$

where the specific enthalpy is defined as $h = e + p/\rho$, fluctuation of which is expressed using temperature so that $\Delta h = c_p \Delta T$. This includes the assumption that specific

enthalpy is a function of temperature only, which is true for an ideal gas and a good approximation for liquids with a low coefficient of thermal expansion (Lampinen, 2010). Thus, the following equation only excludes gases near the critical point, such as steam in a typical steam turbine.

$$\overline{u_j'\rho h'} = \overline{u_j'\rho c_p T'} = \rho c_p \overline{u_j' T'} \tag{2.9}$$

The convection of the kinetic energy contains three fluctuating quantities. Taking $(u_i u_i)$ initially as only one quantity yields

$$\overline{u_j \frac{1}{2}\rho u_i u_i} = \bar{u}_j \frac{1}{2}\rho \overline{u_i u_i} + \frac{1}{2}\rho \overline{u_j'(u_i u_i)'}$$

where the first resulting term can be seen as the convection of kinetic energy

$$\bar{u}_j \frac{1}{2}\rho \overline{u_i u_i} = \bar{u}_j \frac{1}{2}\rho \bar{u}_i \bar{u}_i + \bar{u}_j \frac{1}{2}\rho \overline{u_i' u_i'}$$

while the second term requires a few more steps to unravel

$$u_i u_i = \overline{u_i u_i} + (u_i u_i)' = \bar{u}_i \bar{u}_i + \overline{u_i' u_i'} + (u_i u_i)' \Leftrightarrow (u_i u_i)' = u_i u_i - \bar{u}_i \bar{u}_i - \overline{u_i' u_i'}$$

$$u_i u_i = (\bar{u}_i + u_i')^2 = \bar{u}_i \bar{u}_i + 2\bar{u}_i u_i' + u_i' u_i' \Rightarrow (u_i u_i)' = 2\bar{u}_i u_i' + u_i' u_i' - \overline{u_i' u_i'}$$

$$\frac{1}{2}\rho \overline{u_j'(u_i u_i)'} = \frac{1}{2}\rho \left(\overline{u_j' 2\bar{u}_i u_i'} + \overline{u_j' u_i' u_i'} - \overline{u_j' \overline{u_i' u_i'}} \right) = \bar{u}_i \rho \overline{u_i' u_j'} + \frac{1}{2}\rho \overline{u_i' u_i' u_j'}$$

The average work done by viscous stresses is

$$\overline{u_i \sigma_{ij}} = \overline{(\bar{u}_i + u_i')\sigma_{ij}} = \bar{u}_i \overline{\sigma}_{ij} + \overline{u_i' \sigma_{ij}}$$

The kinetic energy in the local derivative is handled similar to its convection, without the convective velocity. After moving some of the new terms to the right hand side, the RANS equations become

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho \bar{u}_j)}{\partial x_j} = 0 \tag{2.10}$$

$$\frac{\partial (\rho \bar{u}_i)}{\partial t} + \frac{\partial (\rho \bar{u}_i \bar{u}_j)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\overline{\sigma}_{ij} - \rho \overline{u_i' u_j'} \right) - \frac{\partial \overline{p}}{\partial x_i} \tag{2.11}$$

$$\frac{\partial \bar{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{u}_j (\bar{E} + \overline{p}) \right] = \frac{\partial}{\partial x_j}\left(\bar{u}_i \overline{\sigma}_{ij} + \overline{u_i' \sigma_{ij}} - \bar{u}_i \rho \overline{u_i' u_j'} - \bar{q}_j - \rho c_p \overline{u_j' T'} - \frac{1}{2}\rho \overline{u_i' u_i' u_j'} \right) \tag{2.12}$$

The averaged viscous stress tensor $\overline{\sigma}_{ij}$, the heat flux $\bar{q}_j$, and the total energy $\bar{E}$ are

$$\overline{\sigma}_{ij} = \mu \left(\frac{\partial \bar{u}_j}{\partial x_i} + \frac{\partial \bar{u}_i}{\partial x_j} - \frac{2}{3}\frac{\partial \bar{u}_k}{\partial x_k}\delta_{ij} \right) \tag{2.13}$$

$$\bar{q}_j = -\frac{\mu c_p}{Pr}\frac{\partial \overline{T}}{\partial x_j} \tag{2.14}$$

$$\bar{E} = \rho \bar{e} + \frac{1}{2}\rho \bar{u}_i \bar{u}_i + \frac{1}{2}\rho \overline{u_i' u_i'} \tag{2.15}$$

Although for most part we have merely replaced the time accurate quantities with their averaged components, one may note that the energy equation in particular has become much more complicated, with a number of new unknowns. One may also note that one may not close a set of five equations containing 32 variables. It is clear that further simplification is needed before the application. Nevertheless, the problem should first be supplemented with the turbulent density.

2.3 Favre-averaging of the Navier-Stokes equations

Favre (1965) has provided lengthy expressions for averaged Navier-Stokes equations, with the fluctuations of density and fluid properties included. The latter are neglected by FINFLO, NASA's online resource (NASA, 2017c), and presumably NASA's CFL3D code. Therefore, it is taken granted that this negligibility has been thoroughly validated, excluding the fluid property fluctuations from the scope of this thesis. Favre's method B (Favre, 1965) is the one discussed here and usually applied in CFD.

The density fluctuations are considered by primarily averaging conservative quantities. Thus, taking the mass-weighted average $\tilde{f}$ of any quantity f

$$\tilde{f} = \frac{\overline{\rho f}}{\bar{\rho}} \qquad f = \tilde{f} + f'' \qquad \bar{\tilde{f}} = \tilde{f} \qquad \overline{\rho f''} = 0 \qquad \overline{f''} \neq 0 \tag{2.16}$$

that is multiplied by the density in the Navier-Stokes equations leaves the mass-weighted fluctuation f'', average of which is not equal to zero. The quantities in question are the velocity components u_i and the specific internal energy e. The other quantities, density ρ itself, pressure p, and temperature T, are time averaged as previously with Eq. (2.7). This way, averaging a conservative quantity simply yields $\overline{\rho f} = \bar{\rho}\tilde{f}$, and the continuity equation

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial (\bar{\rho}\tilde{u}_j)}{\partial x_j} = 0 \tag{2.17}$$

can be written without additional complications, by employing a different definition of the averaged velocity. In this case the momentum convection has three fluctuating quantities, and requires a couple of steps to unfold

$$\overline{\rho u_i u_j} = \overline{\rho \tilde{u}_i \tilde{u}_j} + \overline{\rho \tilde{u}_i u_j''} + \overline{\rho u_i'' \tilde{u}_j} + \overline{\rho u_i'' u_j''}$$

$$\overline{\rho \tilde{u}_i \tilde{u}_j} = \bar{\rho}\tilde{u}_i\tilde{u}_j \qquad \overline{\rho u_i'' \tilde{u}_j} = \overline{\rho u_i''}\tilde{u}_j = 0 \qquad \Rightarrow \qquad \overline{\rho u_i u_j} = \bar{\rho}\tilde{u}_i\tilde{u}_j + \overline{\rho u_i'' u_j''}$$

The momentum equation becomes

$$\frac{\partial (\bar{\rho}\tilde{u}_i)}{\partial t} + \frac{\partial (\bar{\rho}\tilde{u}_i\tilde{u}_j)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\overline{\sigma}_{ij} - \overline{\rho u_i'' u_j''}\right) - \frac{\partial \overline{p}}{\partial x_i} \tag{2.18}$$

Again, most complications are avoided by the change of convention. The Reynolds stress, now $-\overline{\rho u_i'' u_j''}$, has been altered to include the density fluctuations. Considering that we were still lacking a solution for the "simpler" Reynolds stress, this alteration does not further complicate the matter or affect the need to model the tensor.

Since the Navier-Stokes equations mainly concern conservative quantities, which are transported at mass weighted velocities, and not the primitive quantities, detached from mass, this new convention is physically meaningful. Favre-averaging happens, in a sense, naturally, in implementations that primarily solve conservative quantities. However, the difference must be accounted for in experimental measurements. Ultimately, the quantity measured depends on the method of measurement (Favre, 1965).

One problem does arise in calculating the viscous stress. The stress depends on the rate of deformation, which is a function of velocity, not mass weighted velocity. In other words, Eq. (2.18) contains both $\tilde{u}_i$ as well as $\bar{u}_i$, hidden within $\overline{\sigma}_{ij}$. This discrepancy will not do, as it adds unknowns into the equation. Although Eqs. (2.7) and (2.16) give $\bar{u}_i = \tilde{u}_i - \overline{\rho' u_i'}/\bar{\rho}$, here we simply assume $\bar{u}_i \approx \tilde{u}_i$, as is done in both FINFLO and NASA's online resource (NASA, 2017c). Note that this causes inaccuracy only in the viscous stress term. Fortunately, all this does not affect the definition of skin friction, since the friction is calculated from the velocity profile of a non-turbulent, viscous sublayer, where $u_i = \tilde{u}_i = \bar{u}_i$. The immediate vicinity of a wall is dominated by high molecular friction, keeping fluctuations extremely small. The structure of a turbulent boundary layer will be returned to in section 2.9.

The energy equation is, once again, more complicated than the continuity and momentum equations. Nevertheless, its Favre-averaging follows mostly the same path as the Reynolds-averaging did and reaches a form

$$\frac{\partial \tilde{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\tilde{u}_j(\tilde{E} + \overline{p})\right] = \frac{\partial}{\partial x_j}\left(\tilde{u}_i\overline{\sigma}_{ij} + \overline{u_i''\sigma_{ij}} - \tilde{u}_i\overline{\rho u_i'' u_j''} - \bar{q}_j - c_p\overline{\rho u_j'' T'} - \frac{1}{2}\overline{\rho u_i'' u_i'' u_j''}\right) \tag{2.19}$$

where

$$\tilde{E} = \bar{\rho}\tilde{e} + \frac{1}{2}\bar{\rho}\tilde{u}_i\tilde{u}_i + \frac{1}{2}\overline{\rho u_i'' u_i''} \tag{2.20}$$

As with the momentum equation, the only problem added by the turbulent density lies within the definition of velocity in the viscous stress tensor, slightly reducing the accuracy of the viscous friction work.

The naming convention of these equations is somewhat ambiguous, as the title "RANS turbulence models" generally extends to cover models that cite Favre's work as their basis. To embrace this convention of ambiguity, the abbreviation RANS shall from now on primarily refer to the more accurate Favre-averaged equations. The wide application of RANS as an umbrella term also shows in the name of this chapter.

2.4 Boussinesq approximation and Reynolds analogy

Studying the equations above shows that the turbulent mixing is a result of convection by velocity fluctuations, except for the fluctuating viscous forces. As stated earlier, the aim of RANS is to rid the mathematical description of these difficult to solve fluctuations and only present the mean motion. Thus, the turbulent mixing must be modeled as diffusion like transport of the mean quantities.

Boussinesq approximation assumes that the Reynolds stress tensor $-\overline{\rho u_i'' u_j''}$, or simply τ_{ij}, behaves similar to the molecular viscous stress tensor σ_{ij} in Eq. (2.5) and can be written as a function of some diffusion coefficient and the rate of deformation

$$-\overline{\rho u_i'' u_j''} = \tau_{ij} = \mu_t\left(\frac{\partial \bar{u}_j}{\partial x_i} + \frac{\partial \bar{u}_i}{\partial x_j} - \frac{2}{3}\frac{\partial \bar{u}_k}{\partial x_k}\delta_{ij}\right) - \frac{1}{3}\overline{\rho u_k'' u_k''}\delta_{ij} \tag{2.21}$$

again using the Stokes' hypothesis to handle compressibility without the second coefficient. The last term ensures that $\sum_i \tau_{ii} = \sum_i(-\overline{\rho u_i'' u_i''})$.

Coefficient μ_t is called either the turbulent or eddy viscosity. As a scalar it depicts the mixing equally strong in all spatial directions, making it accurate only for an isotropic turbulence, where the turbulent mixing velocities, and thus the normal Reynolds stresses, are equal in each direction, i.e. $\overline{(u'')^2} = \overline{(v'')^2} = \overline{(w'')^2}$. Subtracting two of the normal stresses using the Boussinesq approximation

$$\tau_{11} - \tau_{22} = 2\mu_t \left(\frac{\partial \bar{u}}{\partial x} - \frac{\partial \bar{v}}{\partial y} \right)$$

yields a clear condition for the its correctness; equal rates of dilation in all directions. For example, a fully developed channel flow, where $\partial\bar{u}/\partial x = \partial\bar{v}/\partial y = \partial\bar{w}/\partial z = 0$, fulfills the condition, but the same cannot be said of developing flows and more complex geometries. The anisotropy of turbulence can be evaluated from the mean rates of strain and vorticity (Wallin et al., 2000), but this thesis concentrates on linear models, where the turbulent mixing is assumed to be roughly isotropic by nature.

Since turbulence transports both momentum and heat with the same convective velocity fluctuations, their apparent rates of diffusive transport should also be directly proportional. Hence, the Reynolds analogy defines the turbulent Prandtl number Pr_t as the ratio of momentum and heat transport by the turbulent fluctuations, leading to an expression similar to the molecular diffusion of heat in Eq. (2.6).

$$c_p \overline{\rho u_j'' T'} = -\frac{\mu_t c_p}{Pr_t} \frac{\partial \overline{T}}{\partial x_i} \tag{2.22}$$

Experiments have shown Pr_t to vary as a function of wall distance, and to some extent even flow conditions and material properties (White, 2006). In the simulations conducted for this work and the benchmark flow $Pr_t = 0.9$ (Diskin et al., 2016), an often used default value.

2.5 Turbulent kinetic energy in two-equation models

The terms added in the Reynolds and Favre-averaged total energies $\bar{E}$ and $\tilde{E}$ in Eqs. (2.15) and (2.20), $\frac{1}{2}\overline{u_i' u_i'}$ and $\frac{1}{2}\overline{\rho u_i'' u_i''}/\bar{\rho}$, respectively, make the foundation of a number of turbulence models. The terms are different definitions of the turbulent kinetic energy k, and looking at the mass-averaged one

$$\bar{\rho} k = \frac{1}{2} \overline{\rho u_i'' u_i''} \tag{2.23}$$

it is quite clear indeed that this is the kinetic energy contained in the velocity fluctuations. Consequently, its square root $\sqrt{k}$ must be proportional to the turbulent mixing velocity, making it a handy quantity in describing turbulence. Note that as a scalar it can only present turbulence as an isotropic phenomenon, but this simplification was already made in the Boussinesq approximation.

The k-based model used here requires a free stream boundary condition for k, which is generalized by nondimensionalizing it as the turbulent intensity I

$$I = \frac{\sqrt{\frac{2}{3}k}}{U} \tag{2.24}$$

where $U = \sqrt{\bar{u}_i^2}$ is the fluid velocity. The intensity thus shows the magnitude of the velocity fluctuations compared to the average fluid velocity.

In models that employ it, the turbulent kinetic energy is given its own transport equation. We start with the material derivative of the kinetic energy

$$\begin{aligned}\frac{\mathrm{D}}{\mathrm{D}t}\left(\frac{1}{2}\rho u_i u_i\right) &= \frac{\partial}{\partial t}\left(\frac{1}{2}\rho u_i u_i\right) + u_j\frac{\partial}{\partial x_j}\left(\frac{1}{2}\rho u_i u_i\right)\\ &= \frac{1}{2}u_i\left[\frac{\partial}{\partial t}(\rho u_i) + u_j\frac{\partial}{\partial x_j}(\rho u_i) + \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j}\right]\\ &= \frac{1}{2}u_i\left[\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) - \rho u_i\frac{\partial u_j}{\partial x_j} + \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j}\right]\end{aligned}$$

which leads to the a of local a derivative and a convection in a conservative form

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{1}{2}\rho u_i u_i\right) + \frac{\partial}{\partial x_j}\left(\frac{1}{2}\rho u_i u_i u_j\right) &= \frac{\partial}{\partial t}\left(\frac{1}{2}\rho u_i u_i\right) + u_j\frac{\partial}{\partial x_j}\left(\frac{1}{2}\rho u_i u_i\right) + \frac{1}{2}\rho u_i\frac{\partial u_j}{\partial x_j}\\ &= \frac{1}{2}u_i\left[\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) + \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j}\right]\end{aligned}$$

In the momentum equation (2.2), taking continuity (2.1) into account, we find that

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) = \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j} + u_i\left(\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u_j)}{\partial x_j}\right) = \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j}$$

and so

$$\frac{\partial}{\partial t}\left(\frac{1}{2}\rho u_i u_i\right) + \frac{\partial}{\partial x_j}\left(\frac{1}{2}\rho u_i u_i u_j\right) = u_i\left[\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j)\right] = u_i\left(\frac{\partial \sigma_{ij}}{\partial x_j} - \frac{\partial p}{\partial x_i}\right)$$

Favre-averaging the result, a procedure similar to the earlier shown Reynolds-averaging of the kinetic energy as a part of the total energy, gives

$$\begin{aligned}&\frac{\partial}{\partial t}\left(\frac{1}{2}\bar{\rho}\tilde{u}_i\tilde{u}_i + \rho k\right) + \frac{\partial}{\partial x_j}\left(\frac{1}{2}\bar{\rho}\tilde{u}_i\tilde{u}_i\tilde{u}_j + \rho\tilde{u}_j k\right)\\ &\quad= \tilde{u}_i\frac{\partial\overline{\sigma}_{ij}}{\partial x_j} + \overline{u_i''\frac{\partial\sigma_{ij}}{\partial x_j}} - \tilde{u}_i\frac{\partial\overline{p}}{\partial x_i} - \overline{u_i''\frac{\partial p}{\partial x_i}} + \frac{\partial}{\partial x_j}(\tilde{u}_i\tau_{ij}) - \frac{\partial}{\partial x_j}\left(\frac{1}{2}\overline{\rho u_i'' u_i'' u_j''}\right)\end{aligned}$$

We may also add $\tau_{ij}\frac{\partial\tilde{u}_i}{\partial x_j} - \tau_{ij}\frac{\partial\tilde{u}_i}{\partial x_j}$, equal to zero, to the right-hand side of the equation. Here we assume that the dissipation like $\tau_{ij}\frac{\partial\tilde{u}_i}{\partial x_j}$ is the rate at which turbulent stress converts mean kinetic energy to turbulent kinetic energy, being a negative source term for the former and positive for the latter. The term is sometimes formulated as a function of vorticity rather than shear, with many models having both shear and vorticity based variations (NASA, 2017c).

Separating the mean and fluctuating parts, and their appropriate source terms $\pm\tau_{ij}\frac{\partial\tilde{u}_i}{\partial x_j}$, yields transport equations for the mean kinetic energy

$$\frac{\partial}{\partial t}\left(\frac{1}{2}\bar{\rho}\tilde{u}_i\tilde{u}_i\right) + \frac{\partial}{\partial x_j}\left(\frac{1}{2}\bar{\rho}\tilde{u}_i\tilde{u}_i\tilde{u}_j\right) = \tilde{u}_i\frac{\partial\overline{\sigma}_{ij}}{\partial x_j} + \tilde{u}_i\frac{\partial\overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}(\tilde{u}_i\tau_{ij}) - \tau_{ij}\frac{\partial\tilde{u}_i}{\partial x_j} \qquad (2.25)$$

and the turbulent kinetic energy

$$\frac{\partial(\bar{\rho}k)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_j k)}{\partial x_j} = \tau_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} + \overline{u_i''\frac{\partial \sigma_{ij}}{\partial x_j}} - \overline{u_i''\frac{\partial p}{\partial x_i}} - \frac{\partial}{\partial x_j}\left(\frac{1}{2}\overline{\rho u_i'' u_i'' u_j''}\right) \tag{2.26}$$

where we are mostly interested in the latter. In order to make the equation more presentable, the viscous stress term is split

$$\overline{u_i''\frac{\partial \sigma_{ij}}{\partial x_j}} = \frac{\partial}{\partial x_j}\left(\overline{u_i''\sigma_{ij}}\right) - \overline{\sigma_{ij}\frac{\partial u_i''}{\partial x_j}}$$

into the conservative flux of work done by the fluctuating viscous stresses and the fluctuating viscous dissipation, respectively. The approximation

$$\overline{u_i''\sigma_{ij}} - \frac{1}{2}\overline{\rho u_i'' u_i'' u_j''} \approx \left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j} \tag{2.27}$$

is applied. These two terms on the left side of Eq. (2.27) are in a conservative gradient form, and are expected to behave as diffusion like turbulent transport. Thus, they can be modeled as a product of a diffusion coefficient and the transported quantity gradient. In Eq. (2.27) σ_k is the Schmidt number for k, an empirical coefficient that relates the turbulent transport of turbulent kinetic energy to that of momentum, similar to how Prandtl number relates the molecular diffusions of momentum and heat.

The pressure gradient seems to be neglected by many early k-based models, such as A.N. Kolmogorov's (Spalding, 1991) and the original $k-\epsilon$ (Jones et al., 1972). White (2006) claims that the gradient was included in L. Prandtl's formulation, but his "Über ein neues Formelsystem für die ausgebildete Turbulenz" (1945) could not be found even in this age and time of limitless information, so the claim cannot be verified here. Nevertheless, many newer models (Wilcox, 2008; Menter et al., 2003) follow the convention of neglecting the pressure here.

Employing the simplifications above, Eq. (2.26) reaches the applicable form

$$\frac{\partial(\bar{\rho}k)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_j k)}{\partial x_j} = \tau_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \bar{\rho}\epsilon + \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{2.28}$$

where the dissipation of turbulent kinetic energy ϵ in

$$\bar{\rho}\epsilon = \overline{\sigma_{ij}\frac{\partial u_i''}{\partial x_j}} \tag{2.29}$$

cannot be solved directly, but must be provided an expression for by any model that utilizes the Eq. (2.28). This has been successfully done by a number of so called two-equation models, which give ϵ, or some comparable quantity, a transport equation of its own.

An important property of Eq. (2.28) is that it contains no laminar terms. In the absence of turbulence all of its terms are always equal to zero, making it unable to predict transition from a laminar to turbulent flow. There are separate transition

models, designed to complement turbulence models that employ Eq. (2.28), but often the problem is simply bypassed by setting the inlet boundary condition of k to some low value. This is also the method applied in this work. In practical terms, this makes the whole computational domain at least slightly turbulent, even if the inlet condition is supposed to represent still fluid with velocity only relative to some moving object. Consequently, the sensitivity to boundary conditions becomes a prominent quality of any model that cannot predict transition.

Superficially it may seem that tackling one unknown, k, has merely created another, ϵ, resulting in no real progress. However, if a solution for ϵ was found, two turbulence quantities of different dimensions, k [$\mathrm{m^2\,s^{-2}}$] and ϵ [$\mathrm{m^2\,s^{-3}}$] would be at disposal. Furthermore, assuming that isotropic turbulence can be adequately described by any two turbulence quantities, the kinematic eddy viscosity ν_t [$\mathrm{m^2\,s^{-1}}$] should be expressible as a function of k and ϵ. Dimensional analysis leads to

$$\nu_t = \frac{\mu_t}{\bar{\rho}} \propto \frac{k^2}{\epsilon} \tag{2.30}$$

This equation would be enough to bring the RANS equations to a closure, on the condition that ϵ can be formulated without introducing new unknowns. The SST $k-\omega$ model used in this work offers a good example of such formulation, although instead of ϵ it solves the specific turbulent dissipation $\omega = \epsilon/(\beta^* k)$, where β^* is a dimensionless empirical coefficient.

It should be noted that even isotropic and fully developed turbulence consists of different eddies over a wide range of size and velocity scales (White, 2006). Therefore, it ought to be too complex of a phenomenon to be sufficiently represented by just two quantities, and yet, the subsequent success of two-equation models has shown that, in many cases, it is not.

2.6 Turbulent pressure work hypothesis

This section presents a simple hypothesis for a more complete turbulent kinetic energy transport equation, but this is not used in the simulations of the present work, as the goal is to verify standard turbulence models.

Ignoring the effects of pressure fluctuations on the turbulent kinetic energy seems like an unnecessary simplification. In the averaged energy equation the turbulent pressure work was considered by including it in the enthalpy convection (2.9), the Favre-averaged version of which reads

$$\overline{u_i(\rho e + p)} = \tilde{u}_i(\bar{\rho}\tilde{e} + \bar{p}) + c_p\overline{\rho u_i'' T'}$$

This expression relies on the assumption that the specific enthalpy and internal energy are functions of temperature only, which is true for ideal gas and a decent approximation for many liquids (Lampinen, 2010). Averaging the internal energy convection and the pressure work separately, and writing the fluctuating internal energy as a product of temperature fluctuations and the specific heat capacity in constant volume c_v, gives

$$\begin{aligned}\overline{u_i(\rho e + p)} &= \tilde{u}_i(\overline{\rho e + p}) + \overline{u_i''(\rho e + p)} = \tilde{u}_i(\bar{\rho}\tilde{e} + \bar{p}) + \overline{\rho u_i''(\bar{e} + e')} + \overline{u_i'' p} \\ &= \tilde{u}_i(\bar{\rho}\tilde{e} + \bar{p}) + c_v\overline{\rho u_i'' T'} + \overline{u_i'' p}\end{aligned}$$

Thus, the fluctuating pressure work can be written as the difference of turbulent convections of enthalpy and internal energy for a perfect gas

$$\overline{u_i'' p} = c_p \overline{\rho u_i'' T'} - c_v \overline{\rho u_i'' T'} = (1 - 1/\gamma) c_p \overline{\rho u_i'' T'}$$

where $\gamma = c_p/c_v$ is the ratio of specific heat capacities in constant pressure and volume. This seems physically meaningful, as it is essentially uniform with the equation of state for ideal gas $p = (1 - 1/\gamma)\rho h$.

The pressure term in the pressure work is now expressed as a function of density and temperature, in a form which could be solved using the Reynolds analogy (2.22). However, the analogy would only consider the pressure, the temperature, and the density fluctuations brought by the turbulent mixing of fluid at different temperatures and ignore those caused by the turbulent compression. To use it we must assume

$$\frac{\partial u_i''}{\partial x_i} \approx 0$$

so that the velocity fluctuations are not strong enough to affect density and thus pressure or temperature. The rule of thumb that flows with a Mach number $Ma < 0.3$ are essentially incompressible sets the condition

$$\sqrt{\overline{u_i''^2}} < 0.3a \;\Rightarrow\; I < \frac{0.3}{Ma} \tag{2.31}$$

for the assumption above. Here a is the speed of sound.

The pressure gradient term in the unsimplified turbulent kinetic energy transport equation (2.26) can be divided into the conservative pressure work and the reversible expansion work done by the velocity fluctuations. The former transports kinetic energy while the latter transforms kinetic to internal energy and vice versa.

$$-\overline{u_i'' \frac{\partial p}{\partial x_i}} = -\frac{\partial}{\partial x_i}\left(\overline{u_i'' p}\right) + \overline{p \frac{\partial u_i''}{\partial x_i}}$$

Applying the Reynolds analogy (2.22) to dispose of turbulent fluctuations, the transport equation for k with pressure included becomes

$$\frac{\partial(\bar{\rho}k)}{\partial t} + \frac{\partial(\bar{\rho}\tilde{u}_j k)}{\partial x_j} = \tau_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \bar{\rho}\epsilon + \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \frac{\partial}{\partial x_i}\left[(1 - 1/\gamma)\frac{c_p \mu_t}{Pr_t}\frac{\partial \overline{T}}{\partial x_i}\right] \tag{2.32}$$

The turbulent pressure work being a function of mean temperature gradient does seem a tad unintuitive, but, as stated earlier, it is still equal to the difference of turbulent convections of enthalpy and internal energy. Therefore, this equation should be applicable whenever the Reynolds analogy is used. The destruction term

$$\bar{\rho}\epsilon = \overline{\sigma_{ij}\frac{\partial u_i''}{\partial x_j}} - \overline{p\frac{\partial u_i''}{\partial x_i}} \tag{2.33}$$

is in this case defined as turbulent dissipation minus expansion work done by the turbulent velocity fluctuations, the total rate at which k transforms to the internal

energy. Earlier we assumed that the velocity fluctuations are not strong enough to do expansion work, but retaining the latter term here goes to show that the condition in Eq. (2.31) limits the use of Reynolds analogy but not the idea of solving turbulent pressure work in general. Since the ϵ in Eq. (2.28) is also assumed to be the total rate of energy transform, any empirical measurements and equations that aim to solve it should also be applicable to the new ϵ, which is merely a more accurate definition of the same quantity.

As the expansion work is in reversible form, it may transform internal energy to kinetic energy, should the divergence of velocity fluctuations be positive. In order to abide by the second law of thermodynamics, that nothing ever goes as it should, one must assume that this rate is never greater than dissipation, that is

$$\overline{\sigma_{ij}\frac{\partial u_i''}{\partial x_j}} \geq \overline{p\frac{\partial u_i''}{\partial x_i}} \;\Rightarrow\; \epsilon \geq 0 \tag{2.34}$$

2.7 Internal and kinetic energies

Auvinen (2017) contributed the detailed flowchart in Fig. 2.2 and the explained equations in Fig. 2.3 for Reynolds-averaged internal and kinetic energies. These provide visual clarification of the different forms of energy transportation and transformation discussed above. Note that for Reynolds-averaging $\overline{u_i'p} = \overline{u_i'p'}$, but for Favre-averaging $\overline{u_i''p} = \overline{u_i''p'} + \overline{u_i''}\bar{p}$, which is why some of the terms differ.

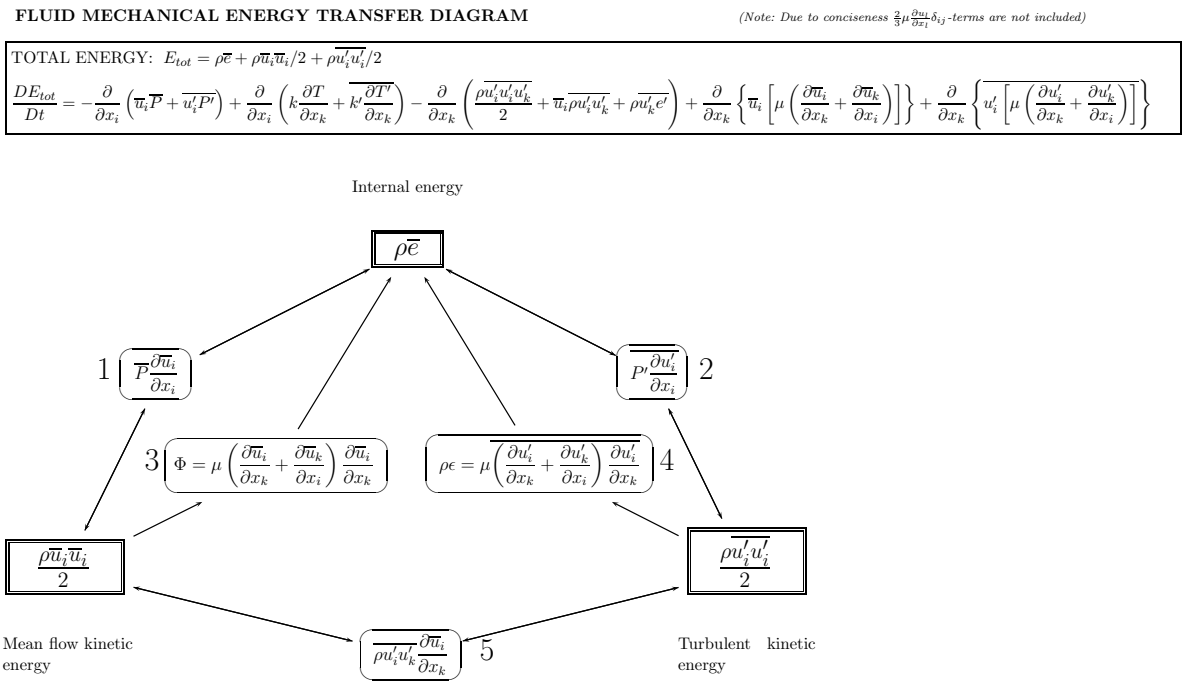

Figure 2.2 Fluid mechanical energy flowchart (Auvinen, 2017).

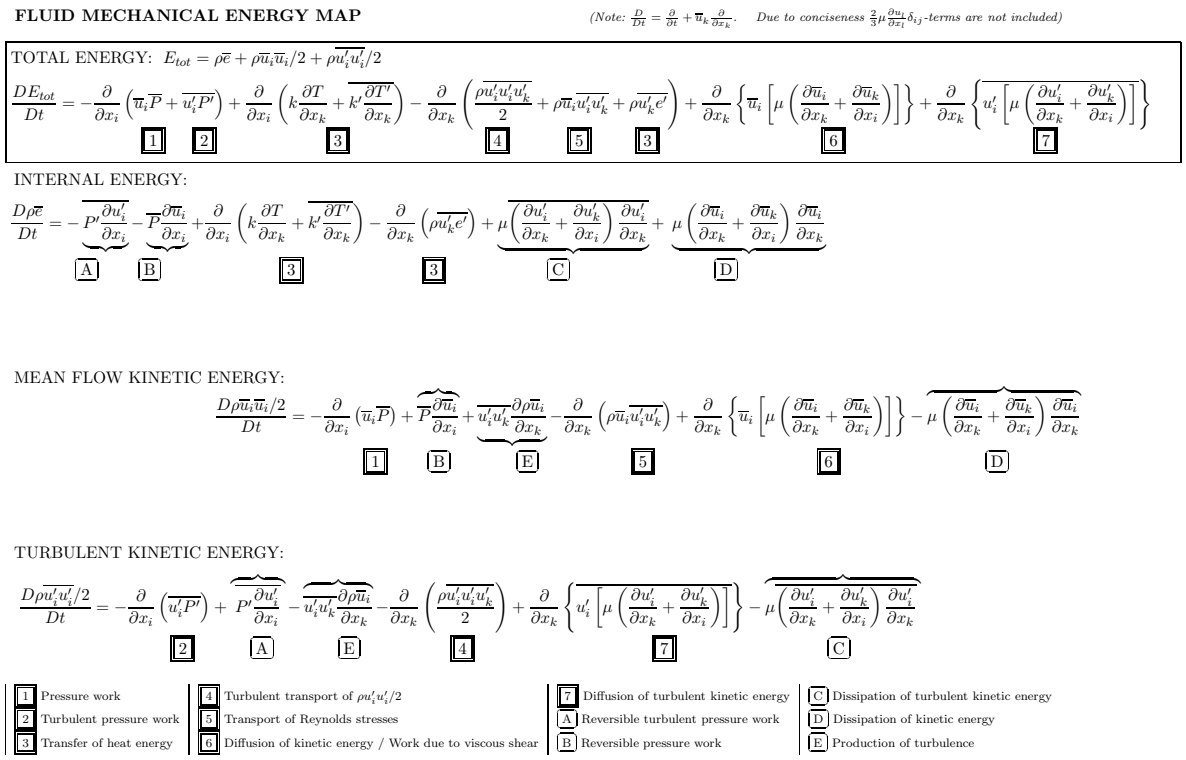

Figure 2.3 Fluid mechanical energy equations (Auvinen, 2017).

2.8 Simplified equations

Applying the approximations and simplifications of Eqs. (2.21), (2.22) and (2.27) to the Favre-averaged Navier-Stokes equations (2.17 - 2.19) yields a more practical expression for the equations to be solved. As these do not include any time accurate quantities or their fluctuations, the tildes and bars indicating averaging may be omitted, as long as we remember that the equations are now written for mean and mass averaged quantities. The simplified RANS equations become

$$\frac{\partial\rho}{\partial t} + \frac{\partial(\rho u_j)}{\partial x_j} = 0 \tag{2.35}$$

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = \frac{\partial}{\partial x_j}(\sigma_{ij} + \tau_{ij}) - \frac{\partial p}{\partial x_i} \tag{2.36}$$

$$\frac{\partial E}{\partial t} + \frac{\partial}{\partial x_j}[u_j(E+p)] = \frac{\partial}{\partial x_j}\left[u_i(\sigma_{ij} + \tau_{ij}) - q_j + \left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{2.37}$$

where the viscous and turbulent stress tensors σ_{ij} and τ_{ij} are written separately, since there is a number of ways to approximate τ_{ij}, but with the Boussinesq approximation (2.21) their sum takes the relatively simple form

$$\sigma_{ij} + \tau_{ij} = (\mu + \mu_t)\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \frac{2}{3}\rho k\delta_{ij} \tag{2.38}$$

The total energy

$$E = \rho e + \frac{1}{2}\rho u_i u_i + \rho k \tag{2.39}$$

has merely dropped the obsolete notations, while the heat flux

$$q_j = -\left(\frac{\mu c_p}{Pr} + \frac{\mu_t c_p}{Pr_t}\right)\frac{\partial T}{\partial x_j} \tag{2.40}$$

now includes the turbulent flux, written using the Reynolds analogy (2.22).

With this formulation, only two new unknowns, μ_t and k, are left to be solved by a turbulence model. Some models, such as the Spalart-Allmaras used in this work, even neglect k and only provide μ_t, still producing good results in their intended range of applications. For example, missing k in the Boussinesq approximation in Eq. (2.38) alters the normal stresses, limiting the use of the model in compressible flows, but has no effect on shear stresses and incompressible flows.

2.9 Turbulent boundary layer

The two turbulence models studied in this work, and many others, make use of the classical turbulent boundary layer equations for a two-dimensional incompressible flow past a smooth wall. That is, they are formulated to give the correct solution for such problem. Furthermore, the important and generalizable observation in the flow is the division of a turbulent boundary layer into four parts from the wall up: viscous, buffer, logarithmic, and outer layers. Another convention is to have three major parts, the inner, overlap, and outer layers (White, 2006), but the former names are used in the papers introducing the Spalart-Allmaras (Spalart et al., 1994) and the original $k-\omega$ (Wilcox, 1988) models and will also be used here.

L. Prandtl deduced that the region very close to a wall is completely dominated by the wall and not significantly affected by the velocity or pressure gradient above the boundary layer (White, 2006). Accordingly, the time-averaged fluid velocity parallel to wall, u, should be a function of just the wall shear stress $\tau_w = \mu(\partial u/\partial y)|_{y=0}$, density ρ, viscosity μ, and wall distance y. At the wall, the no-slip condition must also apply to turbulent fluctuations, resulting in $\mu_t = 0$. Following the Buckingham theorem (Buckingham, 1914), a system of five variables containing three basic dimensions, here the length, time, and mass, can be presented with only two dimensionless variables. These are traditionally formulated as the dimensionless velocity parallel to wall u^+ and the dimensionless wall normal coordinate y^+, defined as

$$u^+ = \frac{u}{u_\tau} \qquad\qquad y^+ = \frac{u_\tau y}{\nu} \tag{2.41}$$

where the so called wall friction velocity

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{2.42}$$

has been defined for convenience.

As the convection, the wall normal velocity, and the pressure gradient are negligible, the momentum equation (2.36) reduces to $\partial^2 u/\partial y^2 = 0$, meaning that velocity profile is linear, shear stress is constant $\mu(\partial u/\partial y) = \tau_w$, and thus

$$u^+ = y^+, \quad y^+ \leq 5 \tag{2.43}$$

This is the linear expression for viscous sublayer, and its thickness $y^+ = 5$ (White, 2006) is an empirical quantity which varies slightly from source to source. The thickness given by White seems to be the minimum for any non-separating flow. This is also the non-turbulent sublayer mentioned earlier which determines the wall friction. Hence, it is important that the computational mesh used is dense enough to cover the sublayer, preferably with $y^+ \leq 1$ at the first row of nodes above a wall (Siikonen, 2014b). Note that this mesh density is the minimum required by a RANS model to work without a wall function, but it does not guarantee numerical accuracy.

The buffer layer following at roughly $5 \leq y^+ \leq 30$ has no analytical solution, but is simply formulated to smoothly join viscous and logarithmic layers when needed.

Moving a little further from the wall, most terms in the momentum equation (2.36) stay negligible, but the turbulent fluctuations become significant. Eventually the turbulent shear will overpower the molecular shear, that is $\mu \ll \mu_t$, and the momentum equation becomes $\mu_t(\partial u/\partial y) = \tau_w$. To solve μ_t, we borrow Prandtl's idea of mixing-length, that the turbulent mixing can be related to some eddy size scale. Furthermore, following Prandtl and T. von Kármán's reasoning that near a wall the scale cannot be greater than the distance to the wall, the turbulent viscosity here can be estimated as the product of some constant and the wall distance y, resulting in $\mu_t = Cy = C\nu y^+/u_\tau$ (Rahman, 2016). Nondimensionalizing the momentum equation leads to

$$\begin{aligned} \mu_t \frac{\partial u}{\partial y} &= \tau_w \\ \frac{C\nu y^+}{u_\tau} \frac{u_\tau^2}{\nu} \frac{\partial u^+}{\partial y^+} &= u_\tau^2 \rho \\ \frac{\partial u^+}{\partial y^+} &= \frac{u_\tau \rho}{C y^+} = \frac{1}{\kappa y^+} \end{aligned}$$

where κ is the Kármán constant, a dimensionless empirical parameter. Integration gives a solution

$$u^+ = \frac{1}{\kappa} \ln y^+ + B \tag{2.44}$$

where $\kappa = 0.41$ and $B = 5.0$ are measured values recommended by White (2006). The equation above fits experimental data well in the region called a logarithmic layer, which begins at about $y^+ = 30$ and ends somewhere at y^+ equal to a few hundred. The thickness of the layer varies widely as a function of pressure gradient. The kinematic eddy viscosity in this sublayer as a function of wall distance after eliminating C becomes

$$\nu_t = \kappa u_\tau y \tag{2.45}$$

and the velocity gradient

$$\frac{\partial u}{\partial y} = \frac{u_\tau^2}{\nu_t} = \frac{u_\tau}{\kappa y} \tag{2.46}$$

both of which are useful in calibrating turbulence models.

The same logarithmic profile of u^+ can be reached by applying functional analysis to join the region close to wall, the inner layer, discussed above, with the outer layer (White, 2006). This does not require any assumptions about turbulent viscosity, but it

does require nondimensionalizing the outer layer, which includes more parameters and is more difficult to solve than the inner layer. While empirical correlations do exist, they are not as important in explaining the turbulence models applied in this work, and, therefore, a close examination of the upper layer is not included here. Nevertheless, it is reassuring to know that multiple paths leading to the Eq. (2.44) exist.

Briefly put, in the upper layer the momentum equation (2.36) does not reduce to only a couple of terms, and u^+ typically rises above the logarithmic curve of Eq. (2.44), as shown in Fig. 2.4. The velocity profile in the upper layer depends heavily on the pressure gradient (White, 2006). The figure also shows agreement to the logarithmic law, but, unfotunately, there is little data available of the extremely thin inner layer. Its existence must then be observed indirectly through various measurements, in addition to the above mentioned deduction that the no-slip condition must result in a viscous sublayer.

Further correlations also exist to account for compressibility in the boundary layer, but, for an adiabatic wall, the differences are insignificant up to $Ma = 5$ (White, 2006). Even when there is a difference, it mostly affects the upper layer, details of which were excluded here, as the models focus on properly forming the viscous, buffer, and logarithmic layers. The models include some calibration for the upper layer, but mostly leave it to the RANS equations. The models employed are individually examined in chapter 4.

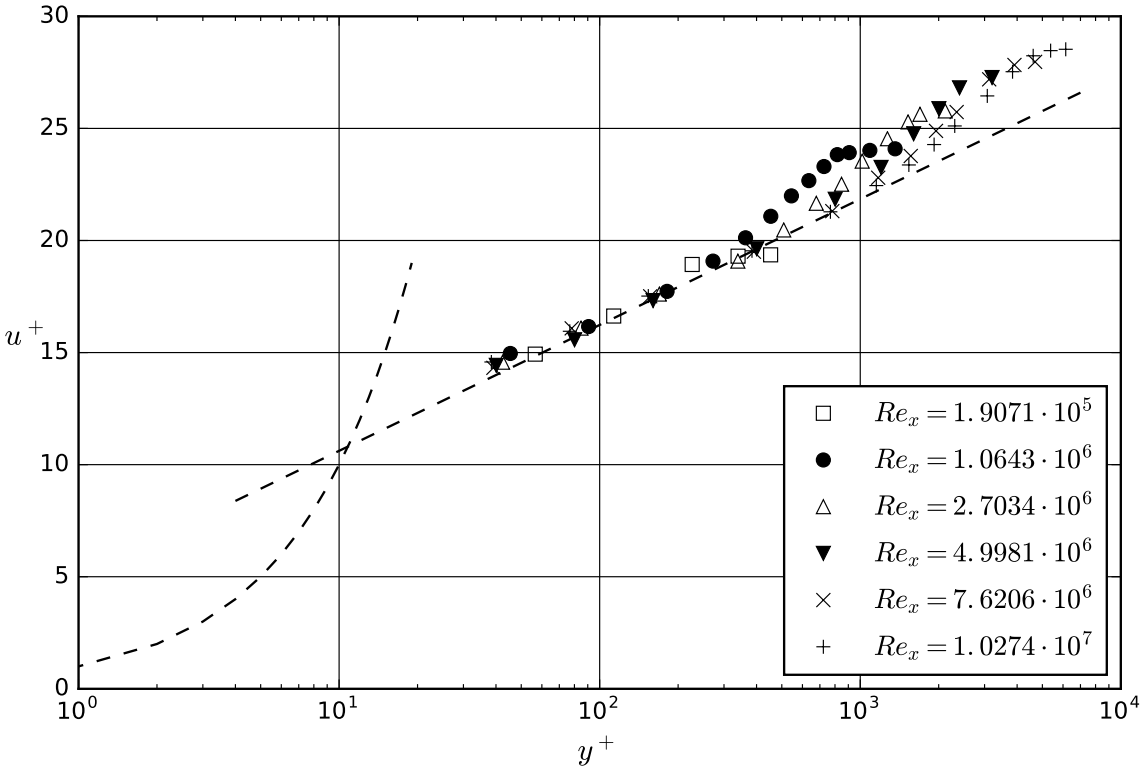

Figure 2.4 Boundary layers measured at different points on a flat plate (Wieghardt et al., 1944; Coles et al., 1968). The variable y^+ is on a logarithmic scale, so the exponential dashed line marks the linear law and the straight line the logarithmic law.

Chapter 3

Verification and validation

This chapter discusses the accuracy of CFD codes and individual solutions alike. The view that these should both be assessed may seem like an obvious remark and a supposed triviality in a master's thesis, rather than its subject, given that an equivalent notion would be just that in many other fields of engineering and applied sciences. Unfortunately, it does not give an optimistic view of the recent progress in this part of CFD that both Roache (1998) and Givi (2016), 18 years apart, see necessary to suggest that more of the computational power newly available should be used to increase the quality of simulations, not only their complexity. The attention the matter has received lately in the form of a special AIAA publication (Givi, 2016) illustrates its importance.

This thesis attempts to follow the terminology presented by Roache (1998), where at least the division of verification and validation, not first proposed by Roache, seems widely accepted. Eloquently defined, verification means ensuring that the equations are solved right and validation that the right equations are solved. Clearly, verification must come first; only if it is known that a set of equations is implemented correctly can their solution be meaningfully scrutinized. Another distinction good to mention is that of error evaluation and estimation: The behavior of an error in a code is evaluated, but a number describing an error in a single solution is estimated.

Verification is a study of whether a numerical solution given by a computer is a good approximation of an analytical solution to the governing equations. It is then a judgment of mathematics and coding, not that of physical modeling (Roache, 1998). Strictly speaking, verification would require an analytical solution, but for turbulent flows in non-trivial geometries, those are not available. If they were, CFD would not be needed. Here we use numerical solutions from a number of other codes instead, relying on the idea that it would be unlikely for all these to have the same programming errors or employ equally insufficient numerical methods to produce similarly flawed results.

Validation, proving that the right equations are solved, requires the right solution, to which the closest thing available is experimental data. Thus, a set of calculations can be validated but a code cannot, and yet, "validation of a CFD code" is in the title of this work. This is to keep the title compact, as it is reasonable to say that a code can be validated for a certain turbulence model and a class of problems (Roache, 1998).

The modern way to categorize error sources in CFD is to have discretization, modeling, and iterative errors (Givi, 2016). The last one comes from solving a set of equations by iterative methods, which are not employed in FINFLO. On the other hand, a category nowadays often thought negligible, the computer round-off error, was found quite relevant. Discussion of discretization error is split in two parts, where the namesake section concerns its evaluation and the following Richardson extrapolation its estimation.

3.1 Discretization error

To solve the set of partial differential equations that is the Navier-Stokes equations, they must be discretized. Discretization means looking for a solution at discrete points in space and time instead of a continuous solution. This allows for a linearisation of the equations into an algebraic form solvable by numerical methods (Siikonen, 2014a).

Fig. 3.1 shows 15 discrete points in a one-dimensional domain, with five spatial points drawn at three different time levels. If l indicates a fixed point in space and n a point in time, any transported quantity T at a discrete point in space and time can be denoted as T_l^n. Customarily, quantities at time level n are known and those at $n+1$ are solved to advance one time step. For example, with the explicit Euler time discretization, a simple diffusion equation

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} \tag{3.1}$$

can now be approximated as

$$\frac{T_l^{n+1} - T_l^n}{\Delta t} = \alpha \frac{T_{l+1}^n - 2T_l^n + T_{l-1}^n}{\Delta x^2} \tag{3.2}$$

where α is some diffusion coefficient.

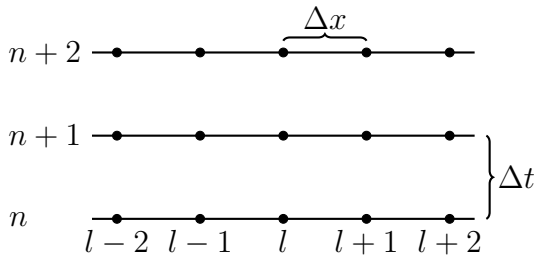

Figure 3.1 One-dimensional differential discretized space at different time levels.

The discretization error is the difference between solutions for discretized and continuous equations (Givi, 2016), such as Eqs. (3.1) and (3.2). One can intuitively tell that as the spatial differences and time steps, Δx and Δt, approach zero, so does the error. More generally, since Δx may not be uniform, let us state that as the number of mesh points N approaches infinity, the error approaches zero. This enables error evaluation, and estimation through observing discrete solutions at different resolutions of discretization. As intuition alone gives no further insight on the connection between the resolution and the error, a more in depth analysis of discretized equations is needed.

As this work only includes steady-state problems where the local change $\Delta T = \partial T/\partial t \Delta t$ vanishes, we may concentrate on discretization in space. While the differential discretized grid in Fig. 3.1 is useful in illustrating that discretization does include defining points in time, for the purpose of explaining the simulations in this thesis, a better figure may be drawn. Fig. 3.2 depicts a grid discretized by finite volumes, with five control volumes, or cells, at one point in time. This cell-centered control volume method results in relatively clear discretized equations and is applied in FINFLO as well as many other codes. For the sake of compactness, only one dimension is shown.

Let us denote a flux component of any conservative quantity T as F_j, the volume of a cell as V, its surface area as S, and a surface component normal to j-direction as S_j.

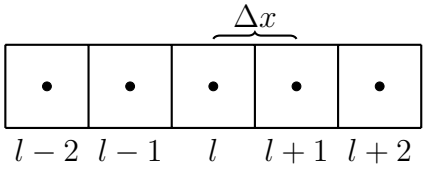

Figure 3.2 One row of cells in a finite volume discretized space.

We also assume that the value of T in the discrete point at the center of a cell is also the average within the cell. Integrating a conservation law in the flux form

$$\frac{\partial T}{\partial t} + \frac{\partial F_j}{\partial x_j} = 0 \tag{3.3}$$

over a control volume l, and applying Gauss's theorem $\int_V \partial F_j/\partial x_j \, \mathrm{d}V = \int_S F_j \, \mathrm{d}S_j$

$$\int_{V_l} \frac{\partial T}{\partial t} \, \mathrm{d}V + \int_{S_l} F_j \, \mathrm{d}S_j = 0$$

results in the discretized flux form

$$V_l \frac{\Delta T_l}{\Delta t} + F_{l+1/2} S_{l+1/2} - F_{l-1/2} S_{l-1/2} = 0 \tag{3.4}$$

where $F_{l\pm1/2}$ are numerical fluxes at the interfaces between two cells and $S_{l\pm1/2}$ are the surface areas of those interfaces. Interpolating or extrapolating the fluxes from discrete points $l \pm 1, 2, 3, ...$ is the source of spatial discretization error in the control volume method, along with the assumption that $\int_{V_l} T \, dV = V_l T_l$.

The momentum equation (2.36) in the flux form of Eq. (3.3) results in $T = \rho u_i$ and

$$F_j = u_j T - (\sigma_{ij} + \tau_{ij}) + \delta_{ij} p \tag{3.5}$$

The flux then has three distinct parts; convection, diffusion and pressure fluxes. The pressure flux in this form is unique in the momentum equation, but the convection and diffusion are common to all transported quantities, including turbulence variables, only with the exception of mass, the diffusion of which is neglected in a homogeneous flow. There is a consensus in the field of CFD that the greatest source of discretization error is the approximation of convection, which generally dictates the magnitude of the error. The magnitude can be evaluated by applying a Taylor series expansion to transform a discretized equation back to continuous form. This is called the modified equation, where the truncated terms form the truncation error of a method. Take for example the simplest possible extrapolation, the first-order upwind method

$$T_{l+1/2} = T_l \tag{3.6}$$

when flow is to the positive direction, left to right in Fig. 3.2, and otherwise $T_{l+1/2} = T_{l+1}$. With this method, ignoring all but the largest truncated term, the modified equation for a discretized $\partial T/\partial x$ in stands (Warming et al., 1974)

$$\frac{\partial T}{\partial x} \approx \frac{T_{l+1/2} - T_{l-1/2}}{\Delta x} = \frac{T_l - T_{l-1}}{\Delta x} = \frac{1}{\Delta x}\left[T_l - \left(T_l - \Delta x \frac{\partial T}{\partial x} + \frac{(-\Delta x)^2}{2}\frac{\partial^2 T}{\partial x^2}\right)\right]$$

$$\frac{\partial T}{\partial x} \approx \frac{\partial T}{\partial x} - \frac{\Delta x}{2}\frac{\partial^2 T}{\partial x^2} \tag{3.7}$$

The largest truncated term, and the main component of the truncation error, is of the order of Δx and is exhibited as $\mathcal{O}(\Delta x)$. The truncation error is only the error of one discretized equation at one discrete point, not the total difference of continuous and discrete solutions, and, hence, not the same as the discretization error. The discretization error also includes any inaccuracies arising from the positioning of discrete points, grid non-uniformity and curvature. The two errors are, however, both functions of the grid resolution, and, thus, connected (Roache, 1998). One may assume that if the grid lines stay the same, but the resolution of discretization varies in a uniform manner across the grid, the two errors are linearly related. To reiterate, if the truncation error is in proportion to Δx^p, where p is the order of the error, the discretization error should be proportional to $1/N^{p/3}$ on a three-dimensional and $1/N^{p/2}$ on a two-dimensional grid in the case of uniform refinement.

Returning to the modified equation (3.7) of the first-order upwind method, multiplying both sides with a convective velocity u shows that instead of only the convection $u(\partial T/\partial x)$, it solves a sum of convection and diffusion with a diffusion coefficient of $u\Delta x/2$. This coefficient is often referred to as the numerical viscosity, and the whole term as the numerical diffusion (Siikonen, 2014a). In general, the truncated terms with even ordered gradients of T cause numerical damping, while the terms with odd ordered gradients create numerical dispersion. The dampening terms all act diffusion like, smoothing out sharp spatial changes, while the dispersive terms do the opposite, escalating them.

Whether an error is dampening or dispersive in nature is important, as dispersion may qualitatively ruin a solution by resulting in an unphysical flow field (Siikonen, 2014a). Although numerical damping helps to attenuate dispersion, ensuring a physically meaningful flow, it is still a numerical error, bringing the discrete solution further from the exact solution. Indeed, the truncation error of the first-order upwind method, which was of $\mathcal{O}(\Delta x)$, has been found far too large for the method to be applicable in practice (Roache, 1998). A higher order method is required, and, interestingly, all second-order accurate upwind biased methods using no more than two points upstream and one downstream can be described by the MUSCL scheme (Van Leer, 1976)

$$T_{l+1/2} = T_l + \frac{1}{4}\left[(1-\kappa)(T_l - T_{l-1}) + (1+\kappa)(T_{l+1} - T_l)\right] \tag{3.8}$$

As with the first-order upwind method (3.6), the equation above is for positive velocities and must be mirrored if convection is to the negative direction. Here κ is a parameter which determines the weight distribution on the discrete points $l-1$, l, and $l+1$.

The modified equation for MUSCL reads

$$\frac{\partial T}{\partial x} \approx \frac{T_{l+1/2} - T_{l-1/2}}{\Delta x} = \frac{\partial T}{\partial x} + \frac{\Delta x^2}{12}(3\kappa - 1)\frac{\partial^3 T}{\partial x^3} + \frac{\Delta x^3}{8}(1-\kappa)\frac{\partial^4 T}{\partial x^4} \tag{3.9}$$

The term proportional to Δx is cut, leaving a largest truncated term of $\mathcal{O}(\Delta x^2)$. In the case of $\kappa = 1/3$, the truncation error will be of $\mathcal{O}(\Delta x^3)$, resulting in a third-order

upwind biased method. This option is used to approximate convection in all simulations executed in this work. "Upwind biased" means that the method considers one point downstream, $l+1$, but puts more weight on the upstream points l and $l-1$. MUSCL can also produce a true second-order upwind scheme with the choice of $\kappa = -1$.

Although the damping term is an order of magnitude below the dispersive term for any $\kappa \neq 1/3$, it is still enough keep the dispersion in check. This property is common to all upwind biased methods (Siikonen, 2014a), making them robust if not necessarily accurate. However, choosing $\kappa = 1$ leads to the central differencing scheme $T_{l+1/2} = (T_l + T_{l+1})/2$. Central differencing is often more accurate than upwind biasing, but it cuts off all damping terms, making the scheme vulnerable to the dispersion. This is not a problem in discretizing diffusion fluxes, since the physical diffusion will always overpower the numerical dispersion, but applying it to convection requires added artificial viscosity to ensure meaningful solutions. This combination of central differencing and artificial viscosity was employed by the CFD code TAU to compute some of the benchmark solutions (Diskin et al., 2016). Other benchmarked codes used the third-order upwind biased method.

Taking a larger number of discrete points to increase the order of accuracy would be a risky endeavor, especially in a three-dimensional grid, since there is no guarantee of, say, $l-2$ actually being upstream of $l+1/2$. To a lesser extent this also applies to $l-1$. Even the third-order method would then only be truly third-order in a one-dimensional case (Siikonen, 2014a). Consider this with the simplification that the order of accuracy p was merely taken from the largest truncated error of discretized convection by determining its magnitude as $\mathcal{O}(\Delta x^p)$, ignoring other contributors to the discretization error. In practice it may not be possible to have a $p > 2$, and there is no guarantee of p being a whole number. The true p can then only be determined by calculating a large series of solutions at different grid resolutions, conducting a grid-refinement study, but when this is not possible, the p deduced above provides the best tool to at least consistently estimate the discretization error (Roache, 1998).

Different methods to increase the order of accuracy do exist, but are not used in this work or widely applied in general. One of the methods, the discontinuous Galerkin, is studied in one of the articles in the special AIAA publication mentioned (Ceze et al., 2016). The discontinuous Galerkin borrows from finite element methods the idea of modeling the transported quantities within a cell using some polynomial, as opposed to the finite volume method, where the quantities are only tracked at cell centers. This allows for a higher order discretization without increasing the number of cells considered.

3.2 Richardson extrapolation

The discretization error being difficult to quantify is a decent reason to only provide a rough estimation, but a poor excuse to provide no estimation at all. The Richardson extrapolation (Roache, 1998; Richardson et al., 1927) approximates the exact analytical solution

$$f_{exact} \cong f_1 + \frac{f_1 - f_2}{r^p - 1} \tag{3.10}$$

from the discrete solutions f_1 and f_2 at two grid densities. Here r is the grid refinement ratio. If the spacing on the coarser grid number (2) is twice that of the finer grid number (1), $r = 2$. This includes the assumption that the solution converges asymptotically with grid refinement. While accepting an exact solution as such would be overoptimistic, this allows for a derivation of the estimated fractional error on grid 1 as

$$E_1 = \frac{f_1 - f_{exact}}{f_1} = \frac{f_2 - f_1}{f_1(r^p - 1)} \tag{3.11}$$

in a way that respects the order of accuracy p, which is also the order of solution convergence. This is not the same as the actual fractional error

$$A_1 = \frac{f_1 - f_{exact}}{f_{exact}} = E_1 + \mathcal{O}\big(\Delta x^{p+m}, E_1^2\big) \tag{3.12}$$

where $m = 1$ for upwind biased and $m = 2$ for central differencing methods. Also, the order p turned out difficult to determine indubitably. Nonetheless, Eq. (3.11) presents a consistent way to estimate the discretization error as E_1. Roache (1998) also proposed to further multiply this by some safety factor to obtain a more conservative estimate called the grid convergence index GCI. For simplicity, in this thesis only E_1 is calculated, equivalent to GCI with the safety factor of one.

3.3 Computer round-off error

The computer round-off error, or machine error, results from a computer only being able to present real numbers up to a certain magnitude and a number of significant digits (Roache, 1998). The limits depend on the number of bits allocated to a variable. Here we are interested in the difference of the standard 32 and 64 bit floating-point numbers.

A 32 bit single precision floating point binary number is composed of a one-bit sign, an eight-bit biased exponent which determines the first significant digit and the order of magnitude, plus a 23-bit trailing significant field (IEEE-754, 2008). The variable then allocates seven bits to the order of magnitude, to a maximum of $2^{2^7} \approx 10^{38}$, which should be enough for any coefficient or quantity, barring some exotic scaling. The significant part, however, converted to the decimal system, is only about seven digits long. The 64 bit double precision number can provide 15 significant decimal digits. This can be illustrated by a line of Fortran code which writes 1/3 in both single and double precision; the former results in 0.3333333 and the latter in 0.333333333333333.

The computer round-off error often being considered negligible presumably means that most modern codes operate in double precision. FINFLO, although arguably modern in the convergence accelerating solver methods implemented, does have its roots in the time when computer memory was precious. Consequently, it only handles the mesh definitions in double precision and stores all flow quantities in single precision variables. A completely double precision variation of the code was compiled to study this error.

3.4 Modeling error

The modeling error is the difference of a solution for the continuous equations describing a flow and a real flow, and so includes errors from sources such as the assumption of

continuity and Newtonian mechanics. When sticking to geometries well above the molecular scale and velocities far below the speed of light, these two errors reduce to absolutely minuscule curiosities. Sources more relevant to practical CFD include the geometric definitions and numerical boundary conditions (Givi, 2016). In this work, however, we trust that the geometry is accurate and, as is possible with external flows, the effects of numerical boundary conditions are diminished by placing the free flow boundaries at a large distance from the simulated object. The most significant source of the modeling error left in the benchmark results is the representation of turbulence.

The previous chapter demonstrated the host of simplifications required to reach a practical form of the RANS equations (2.35 – 2.37). Therefore, the equations could not describe the mean motion with a perfect accuracy even if supposedly correct values for the abstract quantities, such as the turbulent viscosity, were known. The level on uncertainty still grows as these quantities are approximated by a turbulence model. Furthermore, no turbulence model has been proven to work in all situations, but the two models studied here, the Spalart-Allmaras and the SST $k-\omega$, have been shown to often give accurate results in the case of external non-separating flows (NASA, 2017c).

Another source of modeling error evaluated here, although not present in the benchmark results, is the thin layer approximation for viscous fluxes, for which this provides a simplified partial derivative at the interface between two cells at $l+1/2$. The approximation is a faster to solve alternative to full viscous or diffusion terms. The full terms include all the components of the viscous flux F_j^v through all the surface components as S_j. The approximation only considers the presumably largest surface component in each direction, and estimates the partial derivatives at the interfaces as (Siikonen, 2013)

$$\left(\frac{\partial T}{\partial x}\right)_{l+1/2} \approx \frac{2(S_x)_{l+1/2}}{V_{l+1}+V_l}(T_{l+1}-T_l) \tag{3.13}$$

where the distance $\Delta x \approx (V_{l+1}+V_l)/\left[2(S_x)_{l+1/2}\right]$ and $(S_x)_{l+1/2}$ is the x-component of the interface $S_{l+1/2}$. The approximation is accurate if the interface is normal to x-direction, but otherwise adds an error which does not decrease with Δx, and thus is not a part of the discretization error.

3.5 In this work

It was mentioned that as the grid density approaches infinity, the numerical solution converges to the analytical one, which can be observed in a grid refinement study. In a case of a two-dimensional grid, the different densities are achieved by removing every other node in a denser grid to form a sparser one, merging four control volumes into one and doubling Δx_l. The same method applies for three-dimensional grids, where eight control volumes would be merged. In this case, as the solution is presented as the aerodynamic coefficients of a wing profile, we may refer to this approach as the grid convergence of these coefficients. The code is verified by comparing the simulated coefficients to those obtained by other codes, trusting that not all of the codes could accidentally converge the coefficients to the same wrong limit.

The coefficients behaved unexpectedly on the denser grid levels, which was speculated to stem from the use of single precision variables. As stated, a double precision version

of the code was compiled. While a simple compiler option sufficed to produce an executable that ran on a single core, changing the size of the variables going through the message passing interface (MPICH, 2017) to allow for parallel processing proved problematic. The end result, while basically functional, refused to run on as many cores as the original executable. The grid convergence study on the double precision version was then limited from including the densest mesh, on which the solution would have taken more than a month.

The verification concerns the discretization and computer round-off errors plus the implementation of governing equations, anything that deviates the computed solution from the analytical one. The validation evaluates the modeling error and is performed by varying a physical parameter and comparing the results to some experimental data. In this case the parameter was an angle of attack, the angle at which the chord of a wing profile is in relation to the flow.

Roache (1998) warns a CFD practitioner not to blindly trust experimental results and especially not to validate a code against a single experiment. Unfortunately, a single experiment is exactly what is reviewed here. As mentioned, the goal here is not an extensive code validation but an addition to the validated cases. Also, if the implementation of a model is successfully verified, other validation studies performed with a verified implementation of the same model should apply to FINFLO to an extent. It is then perhaps best to think of the validation study included in this thesis as an addition to a global library of validated benchmark flows.

3.6 In philosophy of science

The definitions of verification and validation here are crafted to serve the field of CFD and do contradict those used elsewhere. The contradicting definitions in philosophy of science are particularly problematic, since critical thinking should be applicable, and applied, to CFD, preferably using existing tools provided by the philosophy. These tools help to define science and argue why it should be given weight in decision making. This section aims to provide an example.

Defending CFD as a science helps to justify its application, especially in projects of a public concern, ranging from groundwater models in nuclear waste disposal (Roache, 1998) to climate predictions (Goodwin, 2015). A practitioner must then answer the problem of demarcation, of what is and what is not science. In the two cases mentioned, both authors argue that since any modeling is better than no modeling, and CFD has shown its usefulness before, it ought to be used. This runs into the problem of induction, that no earlier usefulness guarantees universal accuracy, which is why such induction as an answer to the problem of demarcation has been criticized. It would perhaps be better to argue for the scientific nature of CFD using Popper's (1959) famous criterion of falsifiability instead, which the erudite reader is assumed familiar with.

In empirical science, a "philosopher's verification", testing theory against experience, corresponds to our validation. We must stress that CFD is not a single theory, but a sum of its models. The theory we want to validate includes the turbulence models, numerical boundary conditions, and such. Contradictorily, our verification, the judg-

ment of programming and mathematics, is a medley of product testing and numerical analysis, but it is certainly not an application of empirical science.

Roache (1998) does acknowledge the problem of induction by noting that an act of validation only applies to a class of problems. We may also note that CFD cannot be validated outside of a CFD code, making it difficult to falsify; even if the models employed are intended for the class of problems being validated, a false solution may always be pointed on the code used. The code having been verified does not help, even though the results of verification are more universal than those of validation. To say that a verified code always produces a good approximation of an analytical solution would be induction, in which case we would be using induction in an attempt to replace induction as the criterion of science. Popper (1959) also admitted that, excluding empirical methods, it is sometimes perfectly logical to avoid falsification by such refutation of false solutions in his own "third objection" to falsifiability. The question then is whether CFD can be treated empirically.

A CFD code is clearly not an empirical method, although its validation is an application of an empirical science. However, one could employ a number of verified CFD codes to obtain a range of solutions to a benchmark validation case, similar to the benchmark verification case used in this work. This would, in practice, be an empirical method to access the CFD behind the codes and allow for falsification. If different codes consistently gave false solutions for a flow case, one of the models used in the benchmark validation would certainly be false. As the falsification of, say, all turbulence models for an external flow would essentially disqualify the use of CFD in aerodynamics, this procedure would effectively subject CFD to the criterion of falsifiability.

Chapter 4

Flow solution

Three tools are required to solve the discretized RANS equations: An algorithm, a turbulence model, and a computer. One may nitpick over the necessity of the last one, as the earliest research in numerical methods relied on a number of workers calculating by hand (Roache, 1998), but this is hardly a practical approach today.

To begin with listing the computers, the smaller problems in this study were computed on a desktop workstation with the six core Intel Xeon E5-1650 processor and 16GB of RAM, and the larger ones on the Taito supercluster provided by CSC. Taito consists of 983 computational nodes of two different types, the older of which has a pair of eight core Intel Xeon E5-2670 processors and the newer a pair of twelve core Intel Xeon E5-2690v3 processors (CSC, 2017). These are referred to as the Sandy Bridge and Haswell nodes respectively, after the name of processor architecture. The nodes are connected by Infiniband FDR fabric.

This chapter contains an overview of FINFLO and the solution algorithm. Discussion of the algorithm is kept brief, only showing the central idea, as a complete description would serve to double the theoretical background presented in this work. Following the author's interest, the sections concerning the two turbulence models employed are somewhat more in depth, explaining how the turbulent boundary layer influences their formulation.

4.1 FINFLO

FINFLO is a cell-centered, structured finite volume method CFD code, capable of solving LES, DES and RANS equations. Parallel processing is done by dividing a mesh into a number of blocks which trade information explicitly as boundary conditions, allowing one block to be handled by one processor core. In this work Roe's flux-difference splitting (Siikonen, 1995; Roe, 1981) and the MUSCL scheme of Eq. (3.8) with $\kappa = 1/3$ are employed. Viscosity is calculated from the Sutherland's law. Multigrid and local time stepping methods are used for a faster steady state solution (Siikonen et al., 1990). These methods and options were also employed in the CFL3D benchmark simulation (Diskin et al., 2016), making it a good point of comparison. In the simulation presented here, the local time step was set by a local Courant number of 1.9. The calculations were started with the multigrid algorithm, which was turned off nearing the solution to avoid any added error.

FINFLO has a number of solution algorithms, the most suitable of which for this Mach number is the Diagonally Dominant Alternating Direction Implicit (DDADI) (Lombard et al., 1983) algorithm. Being an implicit method, it solves fluxes at time level $n+1$, as opposed to the explicit method, shown for example in Eq. (3.2), where they are solved

at level n. The implicit method is more stable and allows for larger time steps to be used, resulting in faster solution convergence. Since there is no need to use higher order time discretization in steady state flows, we may concentrate on the simple implicit Euler method, which in the discrete flux form of Eq. (3.4) stands

$$V_l \frac{\Delta T_l}{\Delta t} + F^{n+1}_{l+1/2} S_{l+1/2} - F^{n+1}_{l-1/2} S_{l-1/2} - V_l Q^{n+1}_l = 0 \tag{4.1}$$

where the source term Q, which the conservative RANS equations do not have, has been added for turbulence models. The implicit equation has more than one unknown, and to reach the delta-form, where these are all expressed as ΔT_l, the fluxes and source term are linearized as $F^{n+1}_{l+1/2} = F^n_{l+1/2} + (\partial F_{l+1/2}/\partial T)\Delta T_{l+1/2}$, resulting in

$$\begin{aligned} V_l \frac{\Delta T_l}{\Delta t} &+ \frac{\partial F_{l+1/2}}{\partial T} S_{l+1/2} \Delta T_{l+1/2} - \frac{\partial F_{l-1/2}}{\partial T} S_{l-1/2} \Delta T_{l-1/2} - V_l \frac{\partial Q_l}{\partial T} \Delta T_l \\ &= -F^n_{l+1/2} S_{l+1/2} + F^n_{l-1/2} S_{l-1/2} + V_l Q^n_l \\ &= \frac{V_l}{\Delta t} R_l \end{aligned} \tag{4.2}$$

where the explicit terms are moved to the right hand side and denoted as the residual R. In a sense, the algorithm is split in the explicit part of calculating R and the implicit part of solving the Eq. (4.2). Convection in the implicit stage is solved by the first-order upwind method to decrease the number of unknowns, as

$$\frac{\partial F_{l+1/2}}{\partial T} \Delta T_{l+1/2} = \frac{\partial F^+_{l+1/2}}{\partial T} \Delta T_l - \frac{\partial F^-_{l+1/2}}{\partial T} \Delta T_{l+1} \tag{4.3}$$

by forming correct $F^{\pm}_{l+1/2}$ by choosing the upwind points according to characteristic speeds. A higher order spatial discretization may be used in the explicit stage to improve accuracy, as discussed in section 3.1.

Eq. (4.2) can be simplified by defining the difference of outgoing and incoming fluxes

$$\frac{\partial F_{l+1/2}}{\partial T} \Delta T_{l+1/2} - \frac{\partial F_{l-1/2}}{\partial T} \Delta T_{l-1/2} = -L_l \Delta T_l \tag{4.4}$$

where L_l is a difference operator in the l-direction. The implicit stage can be written as

$$\left(1 - \frac{\Delta t}{V_l} L_l - \Delta t \frac{\partial Q_l}{\partial T}\right) \Delta T_l = R_l \tag{4.5}$$

Next, consider two dimensions, where $T_{i,j}$ denotes T at a discrete point (i, j). For brevity, $T_{i,j} = T$, $T_{i+1/2,j} = T_{i+1/2}$, and ignoring the source term for now, one may use approximate factorisation

$$1 - \frac{\Delta t}{V} L_i - \frac{\Delta t}{V} L_j = \left(1 - \frac{\Delta t}{V} L_i\right)\left(1 - \frac{\Delta t}{V} L_j\right) + \frac{\Delta t^2}{V^2} L_i L_j$$

where L_j is the corresponding operator in the j-direction. The last term is discarded, adding discretization error of $\mathcal{O}(\Delta t^2)$, but enabling the implicit stage

$$R = \left(1 - \frac{\Delta t}{V} L_i - \frac{\Delta t}{V} L_j\right) \Delta T \approx \left(1 - \frac{\Delta t}{V} L_i\right)\left(1 - \frac{\Delta t}{V} L_j\right) \Delta T$$

to be further split into two steps

$$\left(1 - \frac{\Delta t}{V} L_i\right) \Delta T^* = R$$
$$\left(1 - \frac{\Delta t}{V} L_j\right) \Delta T = \Delta T^*$$

where Δt is a local time step determined by a local Courant number. The idea is to first solve an intermediate change ΔT^* by only proceeding in the i-direction, and then the final change ΔT by taking ΔT^* as the residual when proceeding in the j-direction. This leads to all equations only having three unknowns, allowing for the use of a quick tridiagonal matrix solver instead of iterative solvers. Thus, the approximation disposes of iterative error and accelerates the solution, but pays for this with a higher discretization error. The error of $\mathcal{O}(\Delta t^2)$ is an order of magnitude below the error of Euler time discretization, which is of $\mathcal{O}(\Delta t)$. It can then be neglected in steady state solutions, but it does mean that some iterative pseudo-time stepping is required in time accurate simulations (Siikonen, 2013). A full three-dimensional implicit equation with a source term is approximated

$$\left(1 - \frac{\Delta t}{V} L_i\right) \left(1 - \frac{\Delta t}{V} L_j\right) \left(1 - \frac{\Delta t}{V} L_k\right) \left(1 - \Delta t \frac{\partial Q}{\partial T}\right) \Delta T = R \tag{4.6}$$

where the third dimension is indexed by k.

This would be enough so solve a single quantity T, and is used for turbulence variables, which are approximated by taking all other quantities as constants. However, the Navier-Stokes equations govern five interdependent quantities, and a code must be able to solve not just T but a vector $\boldsymbol{T}$ multiplied by the Jacobian matrix A of the flux vector $\boldsymbol{F}$

$$\boldsymbol{T} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix} \qquad A = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{T}} \qquad \frac{\partial \boldsymbol{T}}{\partial t} + A \frac{\partial \boldsymbol{T}}{\partial x_j} = 0 \tag{4.7}$$

where $\boldsymbol{F}$ is the flux vector and A its Jacobian matrix. The equation on right represents a primitive form of the original conservation equation for vector $\boldsymbol{T}$. It is then actually another form of Navier-Stokes equations, or RANS equations if $\boldsymbol{T}$ is composed of the time-averaged quantities.

Placing these into Eq. (4.6) would result in a set of equations within a set of equations. A great deal of matrix manipulation is required to avoid this. The Jacobian itself is difficult to solve for conservative quantities, as it includes elements like $\partial(\rho u^2)/\partial(\rho u)$, meaning that $\boldsymbol{T}$ must first be transformed to the primitive variables $\widehat{\boldsymbol{T}} = [\rho \quad u \quad v \quad w \quad e]^T$ (Siikonen, 2013)

$$\frac{\partial \boldsymbol{T}}{\partial t} + A \frac{\partial \boldsymbol{T}}{\partial x_j} = \frac{\partial \boldsymbol{T}}{\partial \widehat{\boldsymbol{T}}} \frac{\partial \widehat{\boldsymbol{T}}}{\partial t} + A \frac{\partial \boldsymbol{T}}{\partial \widehat{\boldsymbol{T}}} \frac{\partial \widehat{\boldsymbol{T}}}{\partial x_j} = M \frac{\partial \widehat{\boldsymbol{T}}}{\partial t} + AM \frac{\partial \widehat{\boldsymbol{T}}}{\partial x_j} \tag{4.8}$$

where M is the transformation matrix from conservative to primitive quantities, and multiplying this with M^{-1} shows that the primitive Jacobian $\hat{A}$ is similar to A

$$\hat{A} = M^{-1}AM \qquad\qquad A = M\hat{A}M^{-1} \tag{4.9}$$

The primitive Jacobian and its eigenvectors can be solved, resulting in a diagonalisation

$$\hat{A} = R\Lambda R^{-1} \tag{4.10}$$

where Λ is a diagonal matrix and the columns of matrix R are the right eigenvectors of $\hat{A}$ (Pitkäranta, 2007). Eq. (4.7) can now be expressed as

$$\frac{\partial \boldsymbol{T}}{\partial t} + MR\Lambda R^{-1}M^{-1}\frac{\partial \boldsymbol{T}}{\partial x_j} = 0 \qquad \Rightarrow \qquad R^{-1}M^{-1}\frac{\partial \boldsymbol{T}}{\partial t} + \Lambda R^{-1}M^{-1}\frac{\partial \boldsymbol{T}}{\partial x_j} = 0$$

$$\frac{\partial \boldsymbol{W}}{\partial t} + \Lambda\frac{\partial \boldsymbol{W}}{\partial x_j} = 0 \tag{4.11}$$

where $\boldsymbol{W}$ in $\partial \boldsymbol{W} = R^{-1}M^{-1}\partial \boldsymbol{T}$ is a vector consisting of the characteristic variables (Siikonen, 2013). As these are independent, that is their Jacobian is the diagonal Λ, their change can be solved in Eq. (4.6) by replacing $(\partial F/\partial T)\Delta T$ with $\Lambda\Delta\boldsymbol{W}$, still only having three unknowns in each equation. Finally, the change of conservative quantities is solved as

$$\Delta \boldsymbol{T} = \frac{\partial \boldsymbol{T}}{\partial \boldsymbol{W}}\Delta \boldsymbol{W} = MR\Delta \boldsymbol{W} \tag{4.12}$$

4.2 Spalart-Allmaras turbulence model

The Spalart-Allmaras (SA) model is designed to be used in aerodynamics, or boundary layer dominated flows, allowing it to be built around the boundary layer. The model aims to simulate turbulence by employing a transport equation for an eddy viscosity like turbulence variable $\tilde{\nu}$ (Spalart et al., 1994)

$$\begin{aligned}\frac{\partial \tilde{\nu}}{\partial t} + u_j\frac{\partial \tilde{\nu}}{\partial x_j} = c_{b1}(1-f_{t2})\tilde{S}\tilde{\nu} - \left(c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right)\left(\frac{\tilde{\nu}}{d}\right)^2 \\ + \frac{1}{\sigma_{\tilde{\nu}}}\left[\frac{\partial}{\partial x_j}\left((\nu+\tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right) + c_{b2}\frac{\partial \tilde{\nu}}{\partial x_i}\frac{\partial \tilde{\nu}}{\partial x_i}\right] + f_{t1}\Delta U^2\end{aligned} \tag{4.13}$$

from which the acutal eddy viscosity is calculated as

$$\nu_t = \tilde{\nu}f_{v1} \tag{4.14}$$

The transport equation resembles other flow equations with its local change, convection, and diffusion terms, with the addition of production and destruction terms much like in the k-equation (2.28). The destruction being a function of the wall distance d is a somewhat unphysical approach resembling algebraic models, but having at least some information of turbulence transported gives the model an advantage over purely algebraic zero-equation models, especially in more complex geometries. The second to last term, gradient squared, is a non-conservative expression with no clear analytical reasoning, borrowed from two-equation models where such terms have shown to improve the results. The final "trip term" $f_{t1}\Delta U^2$ and the function f_{t2} provide a built-in

way to model transition, but these are difficult to use and often not employed (NASA, 2017c), as the user must set the velocity at the point of transition (Spalart et al., 1994).

The formulation more relevant to this work, implemented in FINFLO, ignores the trip term and is written in conservative form as

$$\frac{\partial(\rho\tilde{\nu})}{\partial t}+\frac{\partial(u_j\rho\tilde{\nu})}{\partial x_j}=c_{b1}\tilde{S}\rho\tilde{\nu}-\frac{c_{w1}f_w}{\rho}\left(\frac{\rho\tilde{\nu}}{d}\right)^2+\frac{1}{\sigma_{\tilde{\nu}}}\left[\frac{\partial}{\partial x_j}\left((\mu+\rho\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x_j}\right)+c_{b2}\frac{\partial\tilde{\nu}}{\partial x_i}\frac{\partial(\rho\tilde{\nu})}{\partial x_i}\right] \tag{4.15}$$

The production $c_{b1}\tilde{S}\rho\tilde{\nu}$ is mainly a function of the magnitude of the vorticity W in

$$\tilde{S}=W+\frac{\tilde{\nu}}{\kappa^2d^2}f_{v2} \qquad\qquad W=\sqrt{\frac{1}{2}\left(\frac{\partial u_i}{\partial x_j}-\frac{\partial u_j}{\partial x_i}\right)^2} \tag{4.16}$$

Functions

$$f_w=g\left[\frac{1+c_{w3}^6}{g+c_{w3}^6}\right]^{1/6} \qquad f_{v2}=1-\frac{\chi}{1+\chi f_{v1}} \qquad f_{v1}=\frac{\chi^3}{\chi^3+c_{v1}} \tag{4.17}$$

where

$$g=r+c_{w2}(r^6-r) \qquad\qquad r=\frac{\tilde{\nu}}{\tilde{S}\kappa^2d^2} \qquad\qquad \chi=\frac{\tilde{\nu}}{\nu} \tag{4.18}$$

control the model behavior in different parts of the boundary layer.

Function f_w, multiplying the destruction $c_w/\rho(\rho\tilde{\nu}/d)^2$, is approximatively one in the inner layer but decreases in the outer layer to provide a better fit to experimental data. The other two, f_{v1} and f_{v2}, illustrated in Fig. 4.1, apply in the inner layer, where $\tilde{\nu}$ is of the same order or magnitude or less than ν. This is also true outside the boundary layer, where the functions are simply presumed to have no effect due to the lack of steep gradients (Spalart et al., 1994), an example of how the model is built around the layer. The purpose of f_{v1} is to correctly fit $\tilde{\nu}$ into the viscous sublayer of Eq. (2.43) as ν_t in Eq. (4.14), since, for the sake of numerical robustness, $\tilde{\nu}$ follows the logarithmic law up to the wall where its boundary condition is $\tilde{\nu}=0$. Remember that the logarithmic law for velocity results in a simple linear curve for the eddy viscosity. The term including f_{v2}, $\tilde{\nu}f_{v2}/(\kappa^2d^2)$ in production, is zero both far from the wall where $1/d^2\approx 0$ and in the logarithmic and upper layers where $\tilde{\nu}\gg\nu$, but acts as added production or destruction in the inner layer.

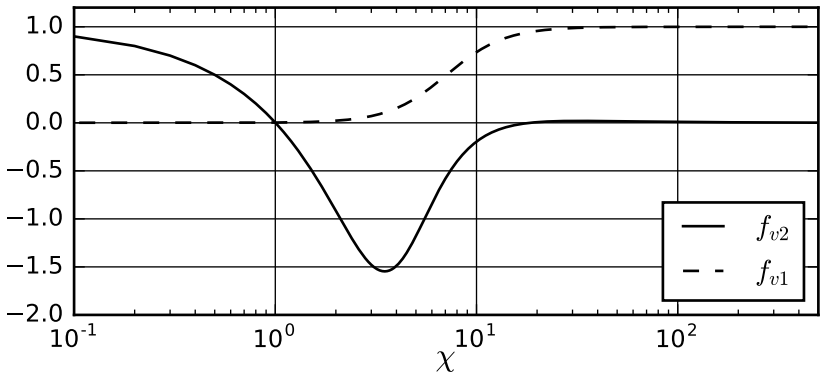

Figure 4.1 Values of f_{v1} and f_{v2} plotted against χ.

In the inner boundary layer, where the only notable gradient is $\partial/\partial y$, $W \approx \partial u/\partial y$, $\mathrm{D}(\rho\tilde{\nu})/\mathrm{D}t = 0$, and ρ is constant, we have

$$c_{b1}\tilde{\nu}\left(d^2\frac{\partial u}{\partial y} + \frac{f_{v2}\tilde{\nu}}{\kappa^2}\right) - c_{w1}f_w\tilde{\nu}^2 + \frac{d^2}{\sigma_{\tilde{\nu}}}\frac{\partial}{\partial y}\left[(\nu+\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial y}\right] + \frac{c_{b2}d^2}{\sigma_{\tilde{\nu}}}\left(\frac{\partial\tilde{\nu}}{\partial y}\right)^2 = 0 \qquad (4.19)$$

In the logarithmic layer of Eq. (2.44), where the eddy viscosity depends linearly on the wall distance $d = y$ so that $d(\partial\tilde{\nu}/\partial y) = \tilde{\nu}$, $\tilde{\nu} = \nu_t = \kappa u_\tau d$, and $\partial u/\partial y = u_\tau/\kappa d$ as in Eqs. (2.45, 2.46), while $\tilde{\nu} \gg \nu$, this further reduces to

$$c_{b1}\tilde{\nu}d^2\frac{\partial u}{\partial y} - c_{w1}\tilde{\nu}^2 + \frac{\tilde{\nu}^2}{\sigma_{\tilde{\nu}}} + \frac{c_{b2}\tilde{\nu}^2}{\sigma_{\tilde{\nu}}} = 0 \qquad \Rightarrow \qquad \frac{c_{b1}d^2}{\kappa u_\tau d}\frac{u_\tau}{\kappa d} - c_{w1}f_w + \frac{1+c_{b2}}{\sigma_{\tilde{\nu}}} = 0$$

$$\frac{c_{b1}}{\kappa^2} - c_{w1} + \frac{1+c_{b2}}{\sigma_{\tilde{\nu}}} = 0 \qquad (4.20)$$

where the constants c_{w1}, c_{b1}, and c_{b2}, plus the Schmidt number $\sigma_{\tilde{\nu}}$, must satisfy the equation for the logarithmic layer to form. The last three can be calibrated in free shear flows and uniform flows with decaying turbulence (Spalart et al., 1994), so this equation effectively sets the correct c_{w1}.

In the viscous and buffer layers the previous assumption of ν_t does not hold, and $\partial u/\partial y = u_\tau^2/(\nu+\nu_t)$, but, as mentioned, $\tilde{\nu}$ should still follow the logarithmic law $\tilde{\nu} = \kappa u_\tau d$. Thus, the contribution of production to Eq. (4.20) must not change, that is

$$c_{b1}\left(\frac{d^2}{\tilde{\nu}}\frac{\partial u}{\partial y} + \frac{f_{v2}}{\kappa^2}\right) = c_{b1}\left(\frac{d}{\kappa}\frac{u_\tau}{\nu+\nu_t} + \frac{f_{v2}}{\kappa^2}\right) = \frac{c_{b1}}{\kappa^2}$$

The reasoning behind f_{v2} is now quite clear; at $d = 0$ $f_{v2} = 1$, and as d increases f_{v2} must decrease until the buffer layer, where it changes direction to compensate for growing ν_t.

The "standard" SA-model as defined by NASA (2017)

$$\begin{aligned}\frac{\partial\tilde{\nu}}{\partial t} + u_j\frac{\partial\tilde{\nu}}{\partial x_j} &= c_{b1}(1-f_{t2})\tilde{S}\tilde{\nu} - \left(c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right)\left(\frac{\tilde{\nu}}{d}\right)^2 \\ &\quad + \frac{1}{\sigma_{\tilde{\nu}}}\left[\frac{\partial}{\partial x_j}\left((\nu+\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x_j}\right) + c_{b2}\frac{\partial\tilde{\nu}}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}\right]\end{aligned} \qquad (4.21)$$

ought to be the one employed in CFL3D for the benchmark solutions (Diskin et al., 2016). Oddly enough, the trip term is ignored here also, but the related f_{t2} is included, and claimed to have no effect (NASA, 2017c). A test implementation of f_{t2} in FINFLO produced clearly false results, so it is likely that in CFL3D $f_{t2} = 0$, even though the website claims otherwise. If so, the only difference would be that FINFLO tries to conserve $\rho\tilde{\nu}$ while CLF3D seems to be in non-conservative form.

It is not very safe to say without experimenting that the conservative form in Eq. (4.15) would do better than Eq. (4.21) in compressible flows, since the model is mostly an empirical fit to an incompressible flow. Also, even the non-conservative form has shown quite good results in a supersonic square duct when coupled with a nonlinear eddy

viscosity scheme instead of the linear Boussinesq approximation (2.21) (NASA, 2017c). This supports the idea that a compressible upper layer may be left to the RANS equations, as long as the model correctly solves an incompressible flow, even if it includes a correlation to modify the upper layer, such as the f_w in SA. Nevertheless, the two versions shown here should not have a noticeable difference in the low-Mach number benchmark flow, since the density varies very little. The practical advantage of the FINFLO formulation is that the convection of $\rho\tilde{\nu}$ can be solved with the same routines as all other conservative convections.

Some of the benchmark solutions were calculated with a model version that changes a behavior at negative $\tilde{\nu}$. Both FINFLO and CFL3D set a minimum limit $\tilde{\nu} \geq 10^{-12}$ to avoid the problem.

Finally, the constants in both FINFLO and the publication (Spalart et al., 1994) are

$$c_{b1} = 0.1355 \quad c_{b2} = 0.622 \quad \sigma_{\tilde{\nu}} = 2/3 \quad c_{w2} = 0.3 \quad c_{w3} = 2 \quad c_{v1} = 7.1 \tag{4.22}$$

4.3 Shear stress transport $k - \omega$ turbulence model

The Shear stress transport (SST) $k - \omega$, like all k-based two-equation models, is more clearly derived from the governing RANS equations than eddy viscosity transporting one-equation models. Unlike ν_t, k has some physical meaning instead of only being an approximation of turbulent mixing. This affects the solution of the energy equation (2.37) especially, where SA ignores k, theoretically making two-equation models much better suited to solve flows dominated by thermodynamics, concerning flows with large density or temperature differences. Nonetheless, the second turbulence quantity in two-equation models, often referred to as the scale determining variable, required to solve turbulent dissipation and viscosity, is nearly as empirical as the $\tilde{\nu}$ of SA. While its transport equation is derivable in some form, the equations used in practice are fairly experimental compared to the k-equation. The scale determining variable is even calibrated much like $\tilde{\nu}$, by ensuring that the boundary layer is formed correctly in an incompressible flow.

The basic $k - \epsilon$ models are unable to correctly shape the inner layer without some added wall distance dependent source terms (Chien, 1982). The $k - \omega$ formulation (Wilcox, 1988), however, does not need any information about walls other than their boundary conditions, and thus seems like a physically more meaningful formulation near the walls. The one disadvantage of the $k - \omega$ approach is its sensitivity to free stream boundary conditions for the turbulence variables, which the $k - \epsilon$, by contrast, handles consistently. The SST $k - \omega$ model (Menter, 1994) attempts to counter this by behaving as a $k - \omega$ model in the boundary layer and a $k - \epsilon$ model outside of it. This runs back into the problem that a point in space must magically know its distance to the nearest wall to determine whether it is in the boundary layer or not. However, the SST model hides the distance within functions that are constant in most parts of the flow and only vary with the distance near the edge of a boundary layer. Away from the edge region the turbulence variables are not affected by the distance, and thus, the model mostly retains its physical approach to turbulence.

The SST-model has a number of variations, of which only the most standard implementation found in FINFLO, with no rotational corrections or such, is presented here. Note that the $k-\omega$ model formulations often break the convention where the Schmidt number σ always divides the eddy viscosity. The σ here multiplies ν_t instead, which should be considered if the same σ_k is used in the energy equation (2.37). The equations read

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho u_j k)}{\partial x_j}=\tilde{P}-\beta^*\rho k\omega+\frac{\partial}{\partial x_j}\left[(\mu+\sigma_k\mu_t)\frac{\partial k}{\partial x_j}\right] \tag{4.23}$$

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho u_j \omega)}{\partial x_j}=\frac{\gamma}{\nu_t}\tilde{P}-\beta\rho\omega^2+\frac{\partial}{\partial x_j}\left[(\mu+\sigma_\omega\mu_t)\frac{\partial \omega}{\partial x_j}\right]+2(1-F_1)\sigma_{\omega 2}\frac{\rho}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \tag{4.24}$$

The k-equation is practically the same as Eq. (2.28), but with a limited production, and a destruction of $\beta^*\rho k\omega$ instead of $\rho\epsilon$. It makes sense that the destruction of k should depend on the quantity itself, and this is also why $k-\omega$ performs well in the inner layer. Both k and ϵ are zero at the wall but ω is not, allowing for a suitable boundary condition to be chosen, as will be shown. The formulation of the ω-equation follows the same pattern as k, with the material derivative equaling production, destruction and diffusion, with the addition of the cross diffusion term in the end. As mentioned with the SA-model, such term has been shown to improve results in $k-\epsilon$-models.

The eddy viscosity is calculated from

$$\mu_t=\frac{a_1\rho k}{\max(a_1\omega, WF_2F_3)} \tag{4.25}$$

where $a_1=0.31$ and W is the absolute value of vorticity. The production of k

$$\tilde{P}=\min\left(\tau_{ij}\frac{\partial u_i}{\partial x_j}, 20\beta^*\rho k\omega\right) \tag{4.26}$$

is limited to 20 times the destruction, in some versions only to 10, to prevent turbulence build-up in stagnation regions (Menter et al., 2003).

Functions F_1 and F_2 control whether the model acts as a $k-\omega$ or a $k-\epsilon$ model, both being equal to one in the boundary layer and zero elsewhere. Parameter F_3 equals to one everywhere but in the viscous sublayer, where it limits μ_t in case of a rough surface (Hellsten, 1998). With smooth surfaces, as in this work, μ_t is always zero in the sublayer and F_3 has no effect. F_1 switches the cross diffusion term off and F_2 limits μ_t to the maximum of $a_1\rho k/W$ in the boundary layer. The latter limitation improves the prediction of flow detachment (Wilcox, 2008). The model also has two sets of constants for β, γ, σ_k, and σ_ω, the inner (1) and outer (2) sets, which are blended as

$$\beta=\beta_1F_1+\beta_2(1-F_1) \tag{4.27}$$

to employ different constants inside and outside the layer, according to the underlying base model $k-\epsilon$ or $k-\omega$.

The functions, plotted in Fig. 4.2, are defined as

$$F_1=\tanh\left\{\min\left[\max\left(\frac{\sqrt{k}}{\beta^*\omega d}, \frac{500\nu}{d^2\omega}\right), \frac{4\rho\sigma_{\omega 2}k}{CD_{k\omega}d^2}\right]\right\}^4 \tag{4.28}$$

$$CD_{k\omega} = \max\left(2\sigma_{\omega 2}\frac{\rho}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}, 10^{-20}\right) \tag{4.29}$$

$$F_2 = \tanh\left[\max\left(\frac{2\sqrt{k}}{\beta^*\omega d}, \frac{500\nu}{d^2\omega}\right)\right]^2 \tag{4.30}$$

$$F_3 = 1 - \tanh\left(\frac{150\nu}{d^2\omega}\right)^4 \tag{4.31}$$

where $\beta^* = 0.09$ and $\sigma_{\omega 2} = 0.856$ are model constants and d the wall distance.

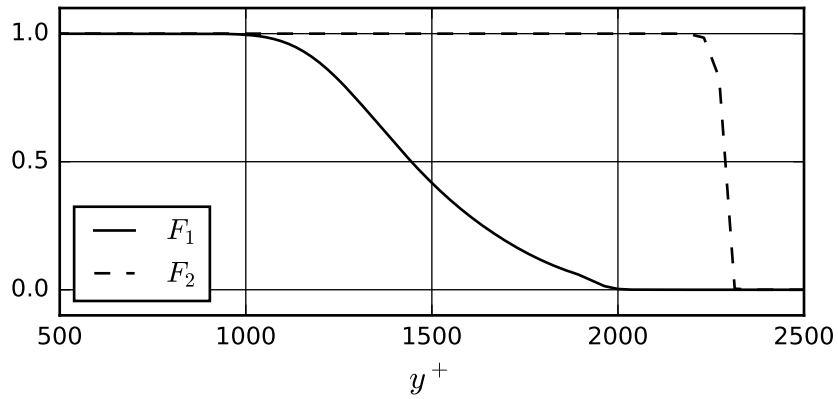

Figure 4.2 Values of F_1 and F_2 plotted against y^+, from a simulation in this work.

The functions are formulated to be continuous instead of just [0, 1]-switches, although the F_2 in this case is almost that steep, and the model is known to sometimes produce a small discontinuity in velocity at the upper edge of boundary layer, where the functions shift (Jantunen, 2016).

To calibrate the model in the logarithmic layer, k [$\mathrm{m^2\,s^{-2}}$] and ω [$\mathrm{s^{-1}}$] must first be given some approximative solutions. If turbulence in that layer is defined by the u_τ [$\mathrm{m\,s^{-1}}$] of Eq. (2.42) and the wall distance $d = y$ [m], dimensional analysis leads to $k \propto u_\tau^2$ and $\omega \propto u_\tau/y$. Realistically, k should also vary with y in the layer, but since k is zero at the wall and close to it in free stream, there must be a point of maximum where $\partial k/\partial y = 0$ somewhere in the boundary layer. As turbulence is destroyed in the inner layer and dispersed in the outer, this point can be assumed to lie within the logarithmic layer. Formulate $k = C_k u_\tau^2$ and $\omega = C_\omega u_\tau/y$, and solve

$$\nu_t = \kappa u_\tau y = \frac{k}{\omega} = \frac{C_k u_\tau^2}{C_\omega u_\tau/y} \qquad \Rightarrow \qquad C_\omega = \frac{C_k}{\kappa}$$

Following the same path as with simplifying the $\tilde{\nu}$-equation in the logarithmic layer, in Eq. (4.19) and below, the k-equation (4.23) reduces to

$$\nu_t \frac{\partial u}{\partial y}\frac{\partial u}{\partial y} - \beta^* k\omega = \kappa u_\tau y \frac{u_\tau^2}{\kappa^2 y^2} - \beta^* C_k^2 \frac{u_\tau^3}{\kappa y} = 0 \qquad \Rightarrow \qquad C_k = \frac{1}{\sqrt{\beta^*}}$$

and the ω-equation (4.24) to

$$\begin{aligned}
\gamma_1 \frac{\partial u}{\partial y}\frac{\partial u}{\partial y} - \beta_1\omega^2 + \sigma_{\omega 1}\kappa u_\tau \frac{\partial \omega}{\partial y} + \sigma_{\omega 1}\kappa u_\tau y \frac{\partial^2 \omega}{\partial y^2} &= 0 \\
\gamma_1 \frac{u_\tau^2}{\kappa^2 y^2} - \beta_1 \frac{u_\tau^2}{\beta^*\kappa^2 y^2} - \sigma_{\omega 1}\kappa u_\tau \frac{u_\tau}{\sqrt{\beta^*}\kappa}\frac{1}{y^2} + \sigma_{\omega 1}\kappa u_\tau y \frac{2u_\tau}{\sqrt{\beta^*}\kappa}\frac{1}{y^3} &= 0 \\
\gamma_1 - \frac{\beta_1}{\beta^*} + \frac{\sigma_{\omega 1}\kappa^2}{\sqrt{\beta^*}} &= 0
\end{aligned} \tag{4.32}$$

which determines the relationship of constants required to form the logarithmic layer. As the other constants can be calibrated elsewhere, this sets the correct multiplier γ_1 for the destruction, similar to the logarithmic layer solution of $\tilde{\nu}$. The outer parameter γ_2 is calculated the same way, with β_2 and $\sigma_{\omega 2}$, even though the equation does not really apply where the outer set 2 constants are used.

The viscous and buffer layers are calibrated by the wall boundary condition of ω, which in the viscous layer should be a function of ν [$\mathrm{m^2\,s^{-1}}$] and y, leading to $\omega = C_\omega \nu / y^2$. The ω-equation here reads

$$\begin{aligned}
\gamma_1 \frac{\partial u}{\partial y}\frac{\partial u}{\partial y} - \beta_1\omega^2 + \nu\frac{\partial^2 \omega}{\partial y^2} &= \gamma_1 \frac{u_\tau^4}{\nu^2} - \beta_1 C_\omega^2 \frac{\nu^2}{y^4} + \nu C_\omega \frac{6\nu}{y^4} = 0 \\
\gamma_1 \left(\frac{u_\tau y}{\nu}\right)^4 - \beta_1 C_\omega^2 + 6C_\omega &= \gamma_1 (y^+)^4 - \beta_1 C_\omega^2 + 6C_\omega = 0
\end{aligned}$$

As $y^+ \to 0$, $C_\omega \to 6/\beta_1$, and in the immediate wall vicinity

$$\omega = \frac{6\nu}{\beta_1 y^2}$$

which is the original smooth wall boundary condition by Wilcox (1988). This results in infinite ω_w, ω at the wall, and must be implemented in the first row of cells above a wall. FINFLO employs a different condition, which accounts for a possible surface roughness and always yields a finite value at the wall, defined as (Hellsten, 1998)

$$\omega_w = \frac{u_\tau^2}{\nu} S_R \qquad S_R = \begin{cases} [50/\max(k_s^+, k_{smin}^+)]^2, & \text{for } k_s^+ < 25 \\ 100/k_s^+, & \text{otherwise} \end{cases} \tag{4.33}$$

where k_s^+ is the nondimensional grain height, calculated from the actual grain hight k_s

$$k_s^+ = u_\tau k_s / \nu \qquad k_{smin}^+ = 2.4(y_1^+)^{0.85} \tag{4.34}$$

and k_{smin}^+ is the value used for smooth surfaces, resulting in the very empirical smooth wall boundary condition

$$\omega_w = 434.03 \frac{u_\tau^{0.3} \nu^{0.7}}{y_1^{1.7}}$$

where y_1^+ and y_1 are the nondimensional and dimensional positions of the first cell center above the wall.

The constants in FINFLO are set as follows.

$$\begin{array}{lllll} a_1 = 0.31, & \beta^* = 0.09, & \beta_1 = 0.075, & \sigma_{k1} = 0.85, & \sigma_{\omega 1} = 0.5 \\ & & \beta_2 = 0.0828, & \sigma_{k2} = 1, & \sigma_{\omega 2} = 0.856 \end{array} \tag{4.35}$$

Chapter 5

Benchmark flow

In recent years, the thought that improved computing capacity should be exploited in verification of CFD codes has led to a number of projects by NASA and AIAA (Givi, 2016). One of the outcomes of these is the special section "Evaluation of RANS Solvers for Benchmark Aerodynamic Flows" in the AIAA Journal of September 2016, volume 54 issue 9. This section, and especially the leading article by Diskin et al. (2016), inspired the FINFLO company to conduct a similar study, resulting in this thesis.

The leading article mentioned reports a grid refinement study on a two-dimensional NACA 0012 airfoil at an angle of attack 10° with specified boundary conditions, defining our benchmark flow. The refinement was done with the SA turbulence model and three CFD codes, all based on the finite volume method: The structured cell-centered CFL3D (NASA, 2017a), the unstructured cell-centered FUN3D (NASA, 2017b), and the unstructured node-centered TAU (DLR, 2017). Structured means that a code can only work with hexahedral control volumes, leading to a simpler high order spatial discretization. Unstructured codes can handle a variety of shapes, allowing for procedurally generated grids to be used. Node-centered means that information is not stored at the center of a cell but at the node points in its corners. The codes then vary in discretization but approximate the same governing equations, with minor differences in the SA model implementation.

The study by Diskin et al. (2016) may be considered a successful verification of the codes, as these all seem to converge the main aerodynamic coefficients to the same limit, despite their differences. Thus, the research realized the hypothesis that different discrete solutions should all approach the same analytical solution as grid spacing nears zero. The primary goal of this thesis is to reproduce this behavior with FINFLO in this benchmark flow, employing the same turbulence model.

Grid refinement was repeated with different options, but due to the limited processor time available, only the primary refinement series includes a solution on the most refined grid level. This is the series employing the original single precision version of FINFLO and the SA model. Other series reach the second densest grid. These series aim to compare the SA and SST models, examine the difference between the thin-layer approximation and the full friction terms in solving the viscous fluxes, and to evaluate the double precision version of FINFLO.

In addition to the grid refinement studies, both SA and SST models were validated by altering the benchmark flow to different angles of attack, following wind tunnel experiments on the airfoil.

5.1 Grid

Diskin et al. (2016) used three different structured two-dimensional high density grids, from which they derived the grid families, I, II, and III, by removing every other node to create a new grid levels. The original grids all have the same dimensions, as consequently do their sparser versions. The levels are listed in the table 5.1.

Table 5.1 The dimensions of different grid levels in number of nodes. $N = N_I \times N_J$

Level	1	2	3	4	5	6
Wall wise nodes N_I	7169	3585	1793	897	449	225
Wall normal nodes N_J	2049	1025	513	257	129	65

The grid families, available at (NASA, 2017c) and shown in Fig. 5.1, have two different properties. The easier one to see is the smoothness with which the different zones, the curvilinear and the right-angled parts, connect. Family I is clearly the most refined in this aspect. The less obvious property is the grid density distribution; family II is the sparsest in the middle of the airfoil, but the most dense near the trailing edge.

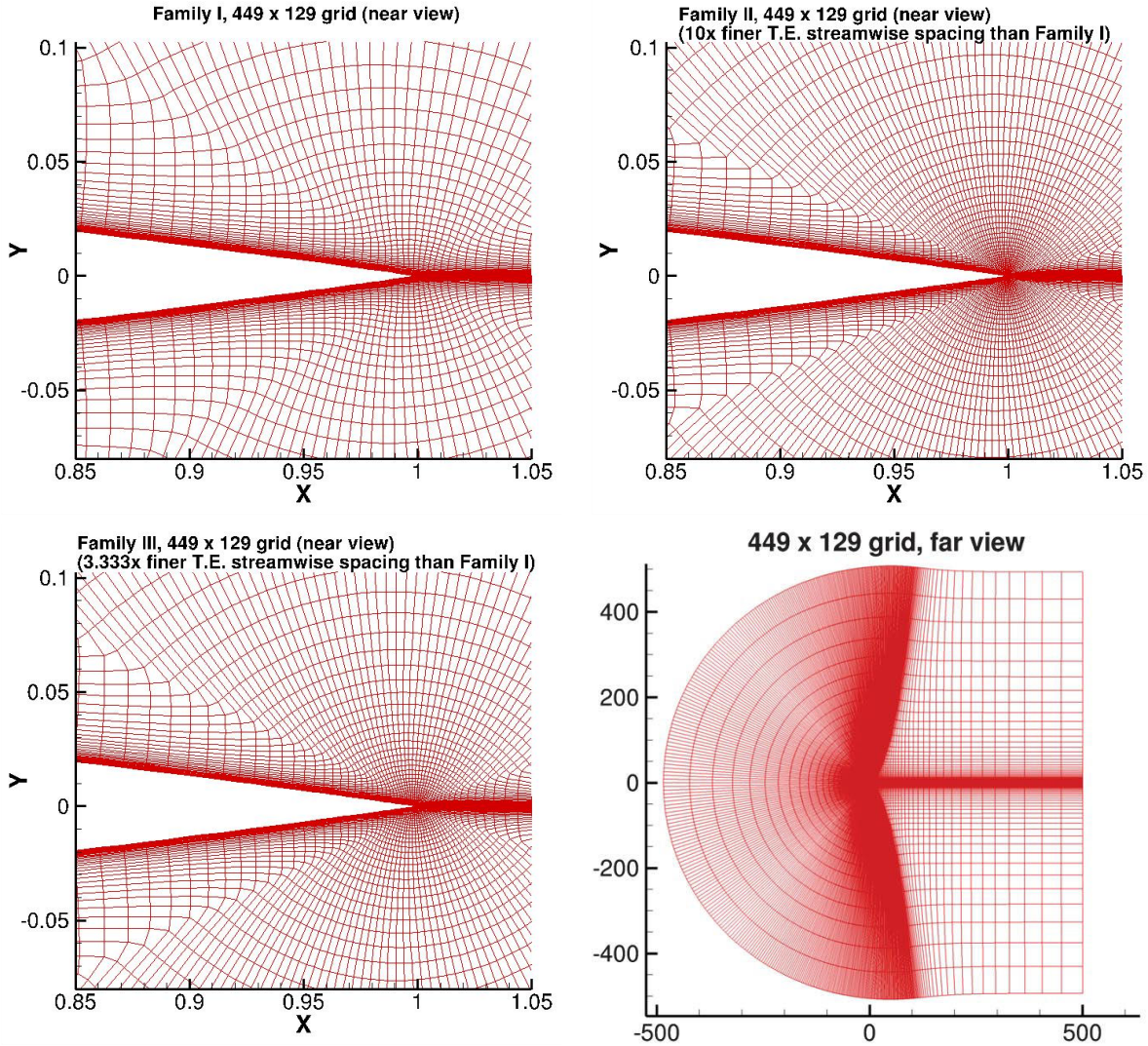

Figure 5.1 The three grid families at level five zoomed on the trailing edge plus the whole domain on bottom right (Diskin et al., 2016).

It was found out that in this case a grid density distribution favoring the trailing edge was more important than the seamless joining of different grid zones, and the family II demonstrated much better grid convergence of the aerodynamic coefficients than the other families (Diskin et al., 2016). Hence, only the family II, drawn around the whole airfoil in Fig. 5.2, is used in the simulations here.

The geometric definition of NACA 0012 employed differs slightly from the original. The standard airfoil has a blunt end, while the one used by Diskin et al. (2016) has a sharp trailing edge. The reason is to test the behavior of a code around a geometric near-singularity. The same non-standard geometry is used in this work.

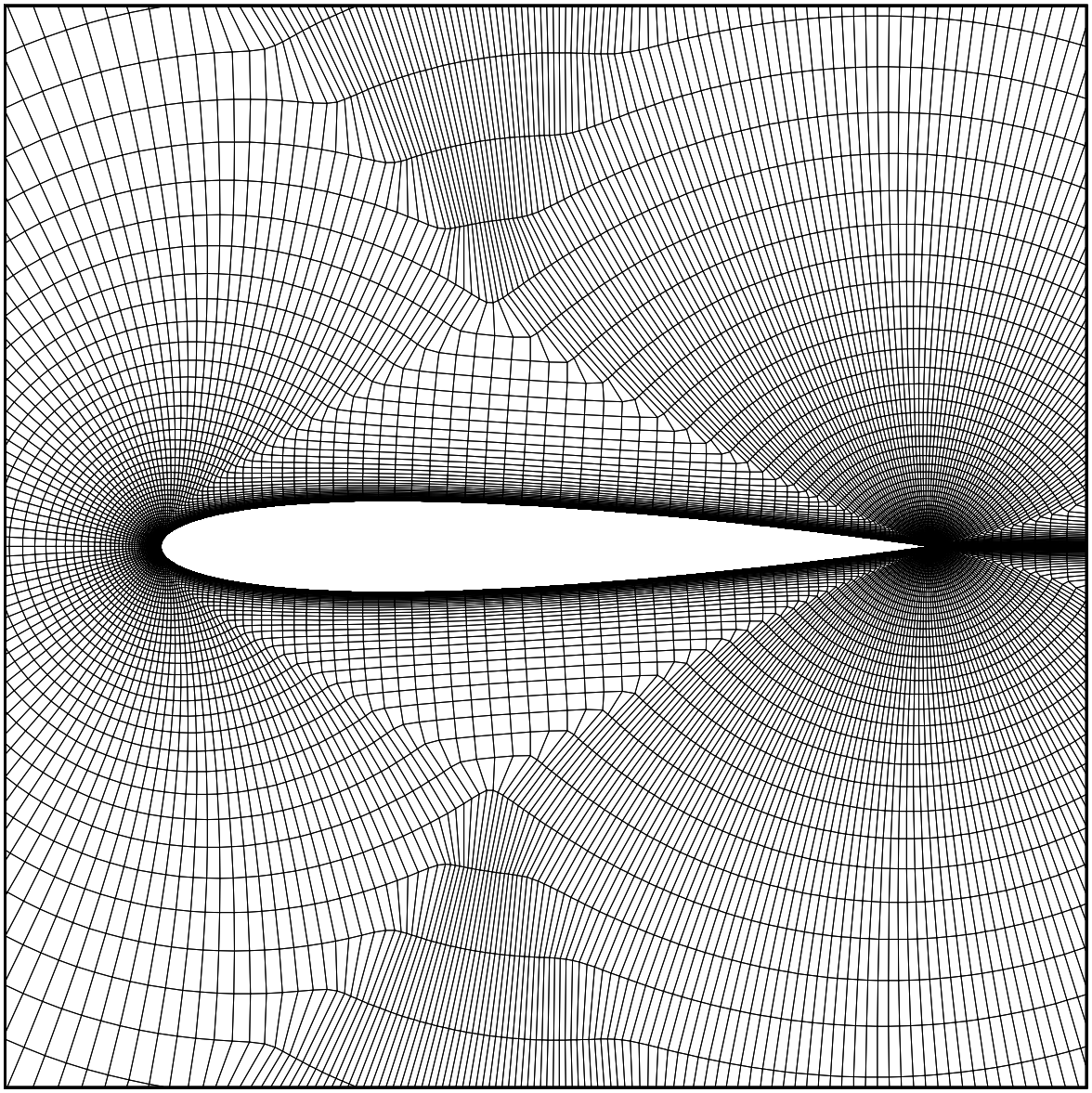

Figure 5.2 Family II level 5 grid around the airfoil.

As noted earlier, the wall normal spacing of a grid must be accurate enough for the viscous sublayer, which usually requires $y^+ \leq 1$ at the center of the first cell off the wall. The grid levels up to five fulfill the condition as shown in Fig. 5.3, but at level six the y^+ near both the leading and trailing edges exceeds one by some margin.

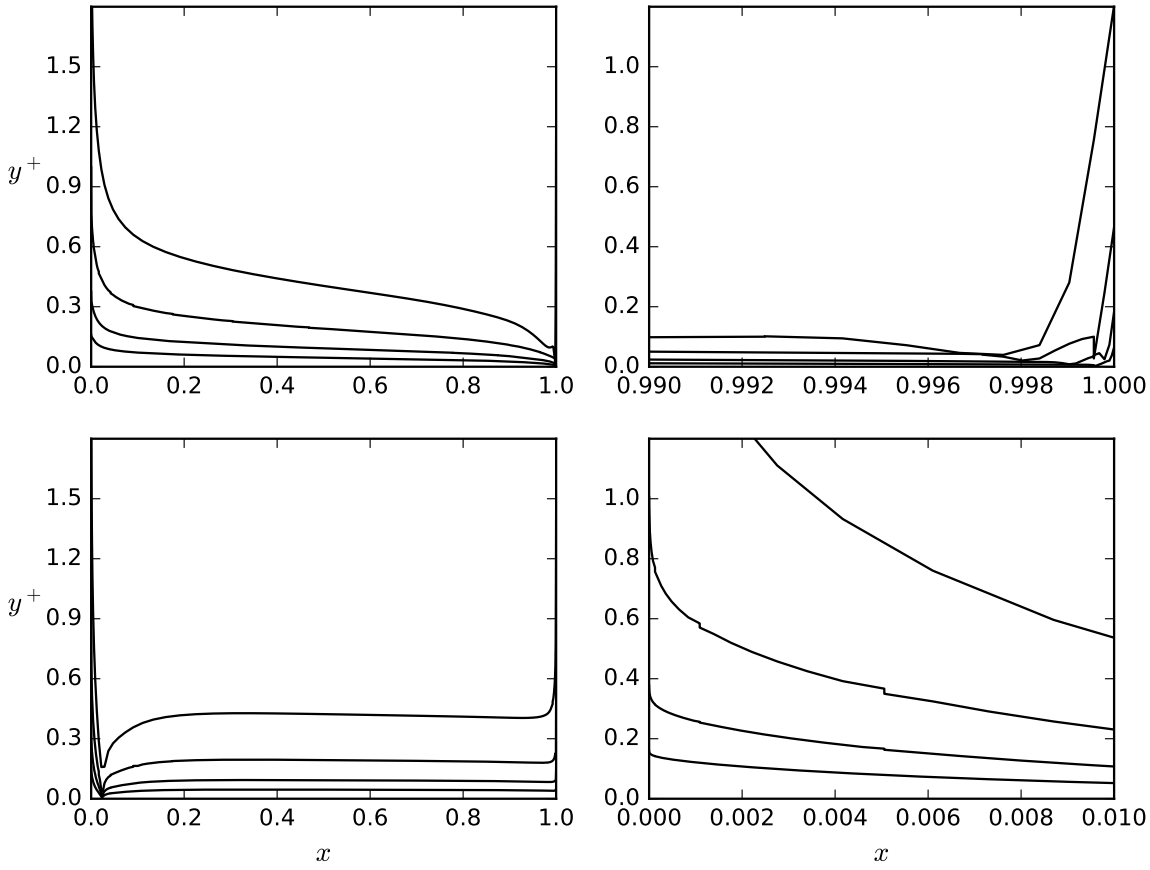

Figure 5.3 Value of y^+ at the center of the first cell off the surface, computed by FINFLO. Upper surface on top row and lower on bottom. Highest curve is the y^+ on grid level 6, under which are values on grid levels 5, 4, and 3.

The small discontinuities seen in y^+ of level five result from the explicit boundaries in between computational blocks, which affect the wall friction and thus y^+. The grid refinement study started with dividing the grid in 64 blocks and solving the flow at level 5. This solution could then be used as the initial guess at level four, and so on, allowing the achievement of level one solution to be accelerated by utilizing both the previous solutions and 64 cores. The discontinuities at level five are small enough to not significantly disturb the surface integral determining the final coefficients, and they further diminish on lower levels. The flow at level six was solved on an undivided grid and one core, avoiding the problem there.

5.2 Boundary conditions

The free-stream flow is defined as $Ma = 0.15$, $T = 300\,\mathrm{K}$, and $Re_L = 6 \cdot 10^6$, where the reference length L is the the airfoil chord length. FINFLO implements these by calculating constant boundary conditions on the inflow surfaces and setting a zero pressure gradient on the outflow surfaces in this subsonic flow. The other codes used the far-field Riemann boundary conditions instead, but since the external surfaces are at $500L$ from the airfoil in the grid (Diskin et al., 2016), this should make very little difference. The angle of attack is $\alpha = 10°$. The wall is an adiabatic surface with a no-slip condition.

As neither of the turbulence models employed can predict transition, non zero boundary conditions for the turbulence quantities were needed. The SA turbulence variable in the free flow is set as $\tilde{\nu}_\infty = 3\nu_\infty$, where the subscript ∞ marks a free flow value. This results in a ratio of eddy and molecular viscosities $\nu_{t,\infty}/\nu_\infty = 0.210438$. The same ratio is used to determine ω_∞ for the SST model, after k_∞ is calculated from the turbulent intensity in Eq. (2.24) by choosing $I_\infty = 0.0001$.

5.3 Grid convergence

The benchmark grid convergence results are shown in Fig. 5.4 by plotting the drag coefficient C_D, the lift coefficient C_L, the pitching moment coefficient C_M at $x = L/4$, and the pressure drag coefficient C_{Dp} against the grid spacing $\sqrt{1/N} \propto \Delta x$. FUN3D and TAU solved the convection of $\tilde{\nu}$ only with the first-order upwind scheme, while CFL3D also calculated a series with a second-order approximation. Both CFL3D series are plotted. FINFLO used the same third-order upwind biased scheme for the convection of $\rho\tilde{\nu}$ as for all other quantities. CFL3D, FUN3D and TAU all used full friction terms.

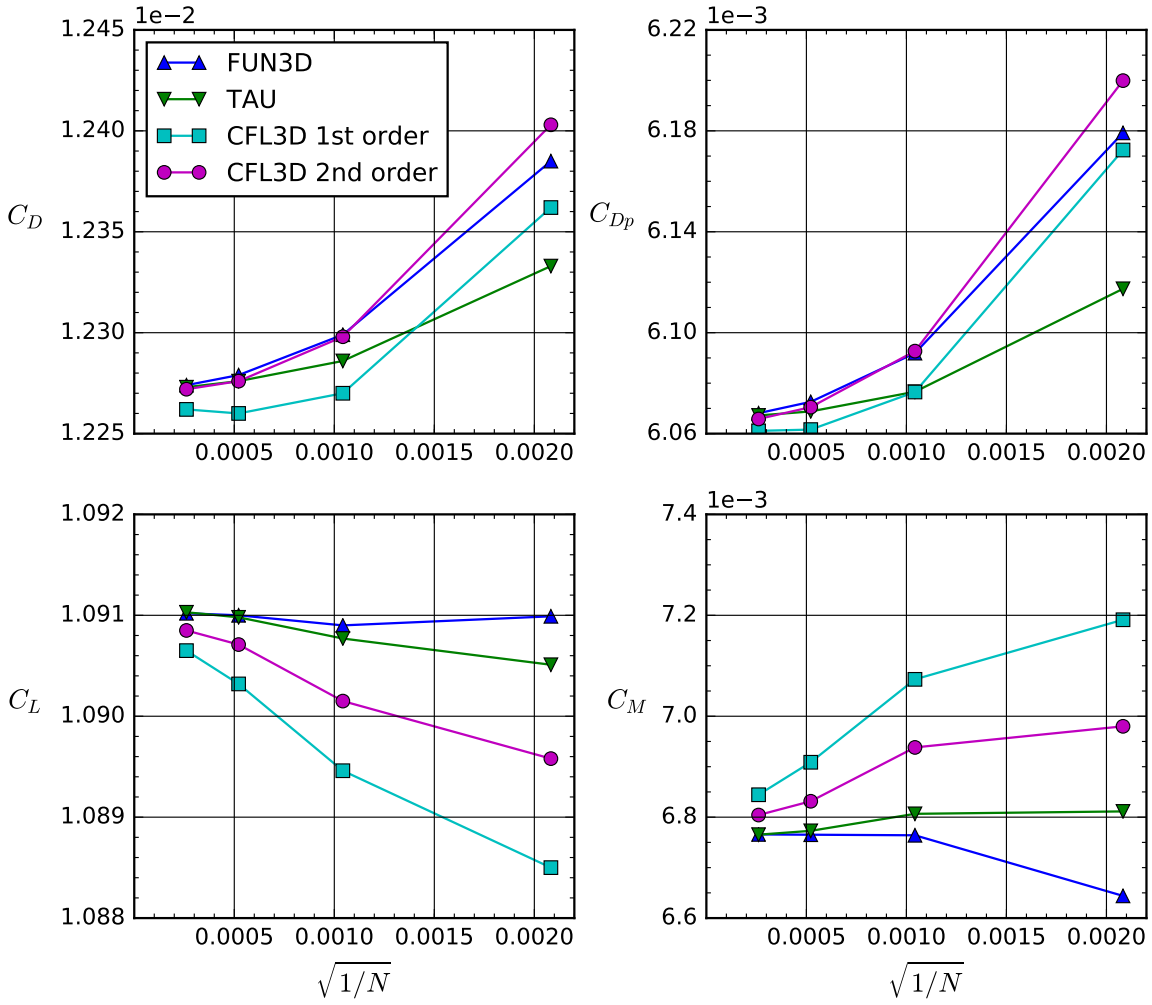

Figure 5.4 The benchmark grid convergence results (Diskin et al., 2016).

The SA model in FINFLO had never seen much use until this work, and the very first results showed the importance of verification. A closer examination revealed that the

minimum value of $\tilde{\nu}$ was incorrectly calculated by a routine meant for the minimum of ϵ. This was fixed by setting $\tilde{\nu}_{\min} = 10^{-12}$, the same as in CFL3D. The coefficients of Fig. 5.4 plus the viscous drag C_{Dv} given by FINFLO using the fixed model are plotted with the CFL3D results in Figs. 5.5 and 5.6. "Thin" marks the results with a thin layer approximation and "double" the double precision version.

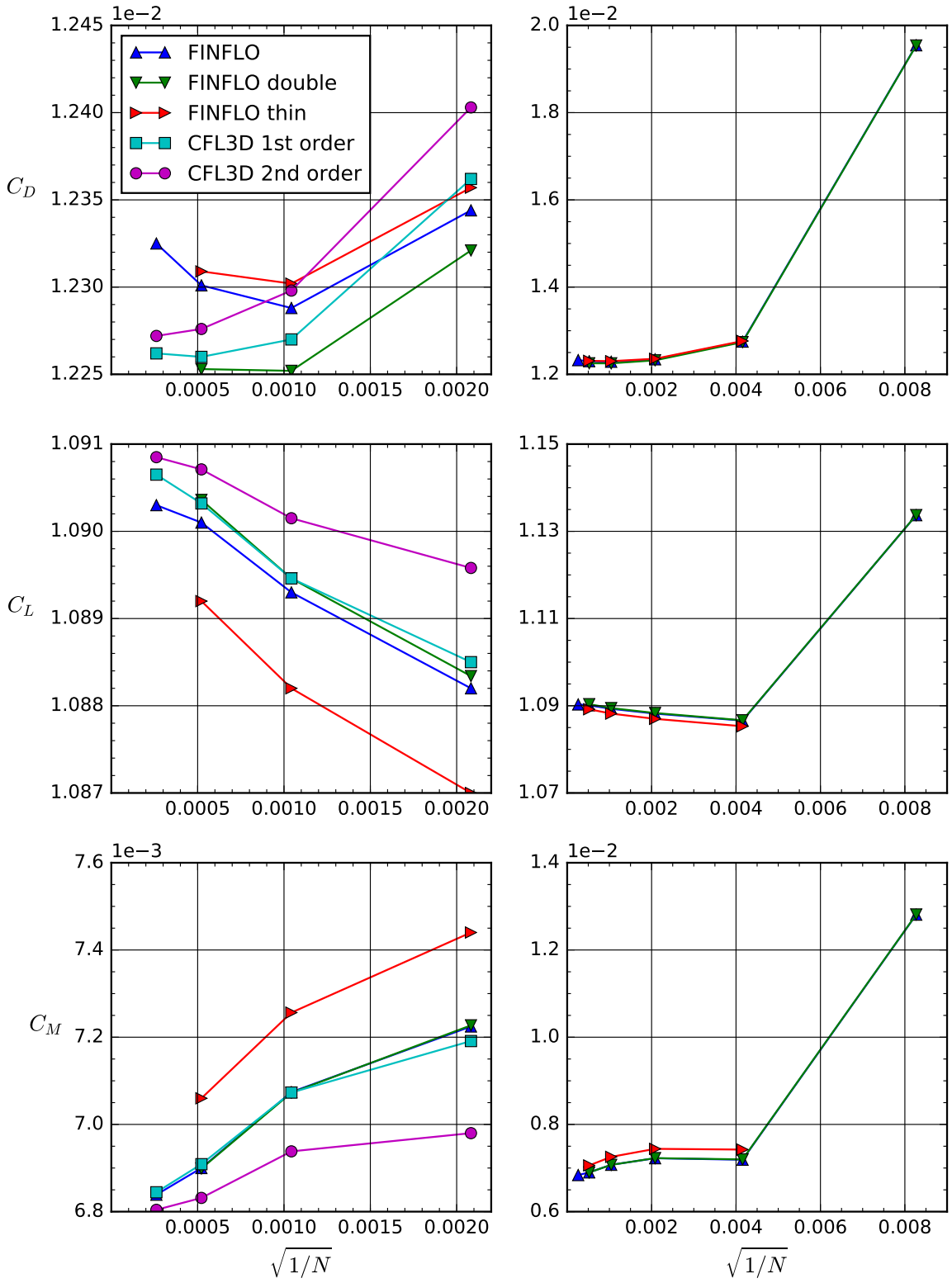

Figure 5.5 Grid convergence of aerodynamic coefficients with SA.

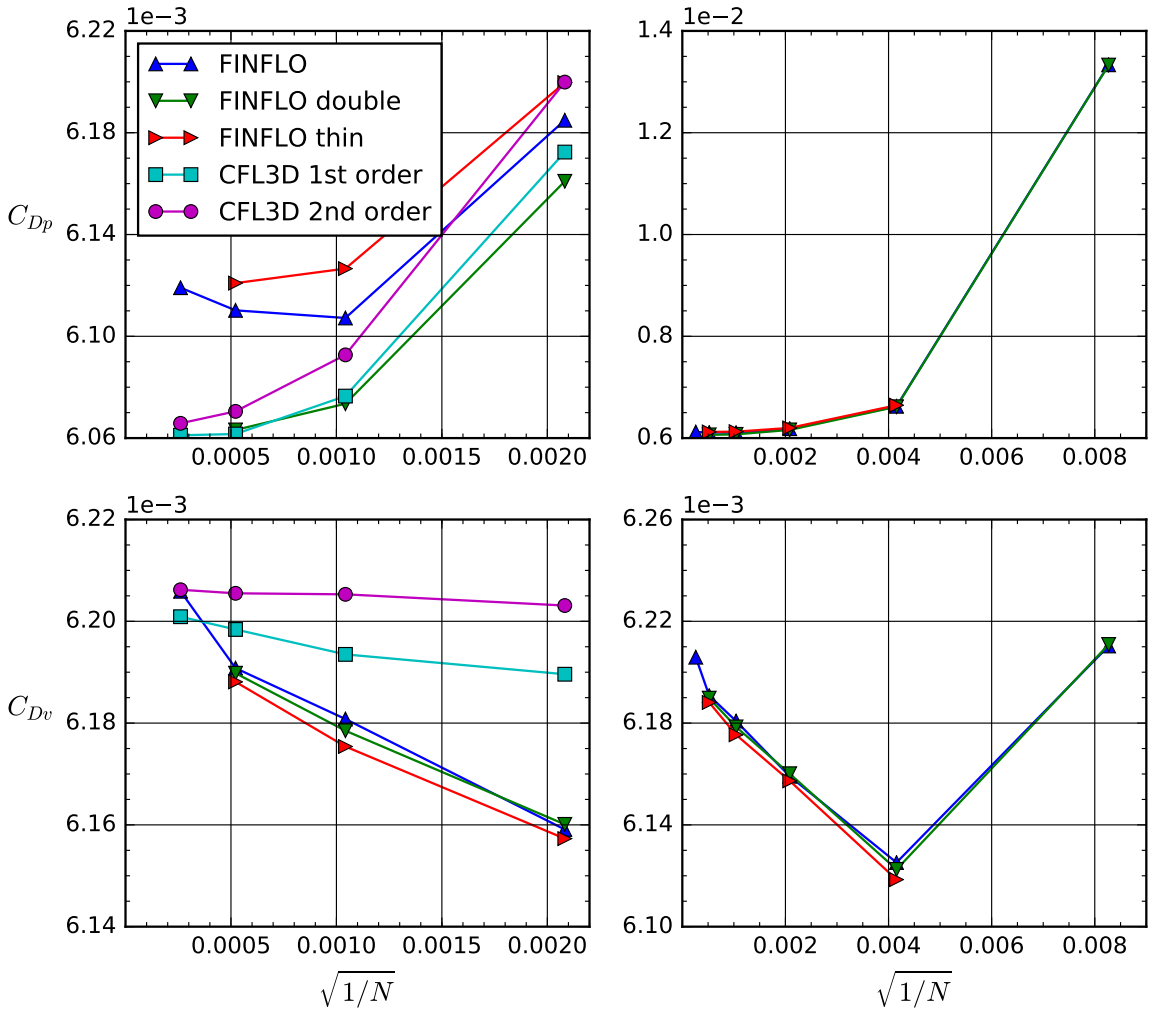

Figure 5.6 Grid convergence of pressure and viscous drag with SA.

The results can also be found as a table in the Appendix. The single precision version of FINFLO does not converge the drag asymptotically but behaves unexpectedly on the most refined levels instead. Surprisingly, the lift and the pitching moment converge much better, even though it is the pressure drag that causes the problem, while viscous drag converges in single precision as well as in double. One would think that if the pressure force were more sensitive to the machine accuracy than the viscous force, this would mostly affect the two coefficients more dependent on the pressure, C_L and C_M, not C_D, half of which is caused by the viscous friction C_{Dv}. It is also quite odd that the thin layer approximation affects the pressure drag more than the friction, but at least it is consistent in that it also alters C_L and C_M more than it does C_D.

The double precision version produces coefficients extremely close to those given by the 1st-order CFL3D, with the exception of C_{Dv}. Both the first-order discretizations of $\tilde{\nu}$ by CFL3D and third-order one by FINFLO seem to approach the same limit, although FINFLO does so slower. This probably means that both codes solve the SA equation correctly, but the third-order approximation in this case is less accurate than first-order, and the numerical diffusion of $\tilde{\nu}$ is beneficial to the model.

The same coefficients computed with the SST model, with both full friction terms and the thin layer approximation, can be found in Figs. 5.7 and 5.8.

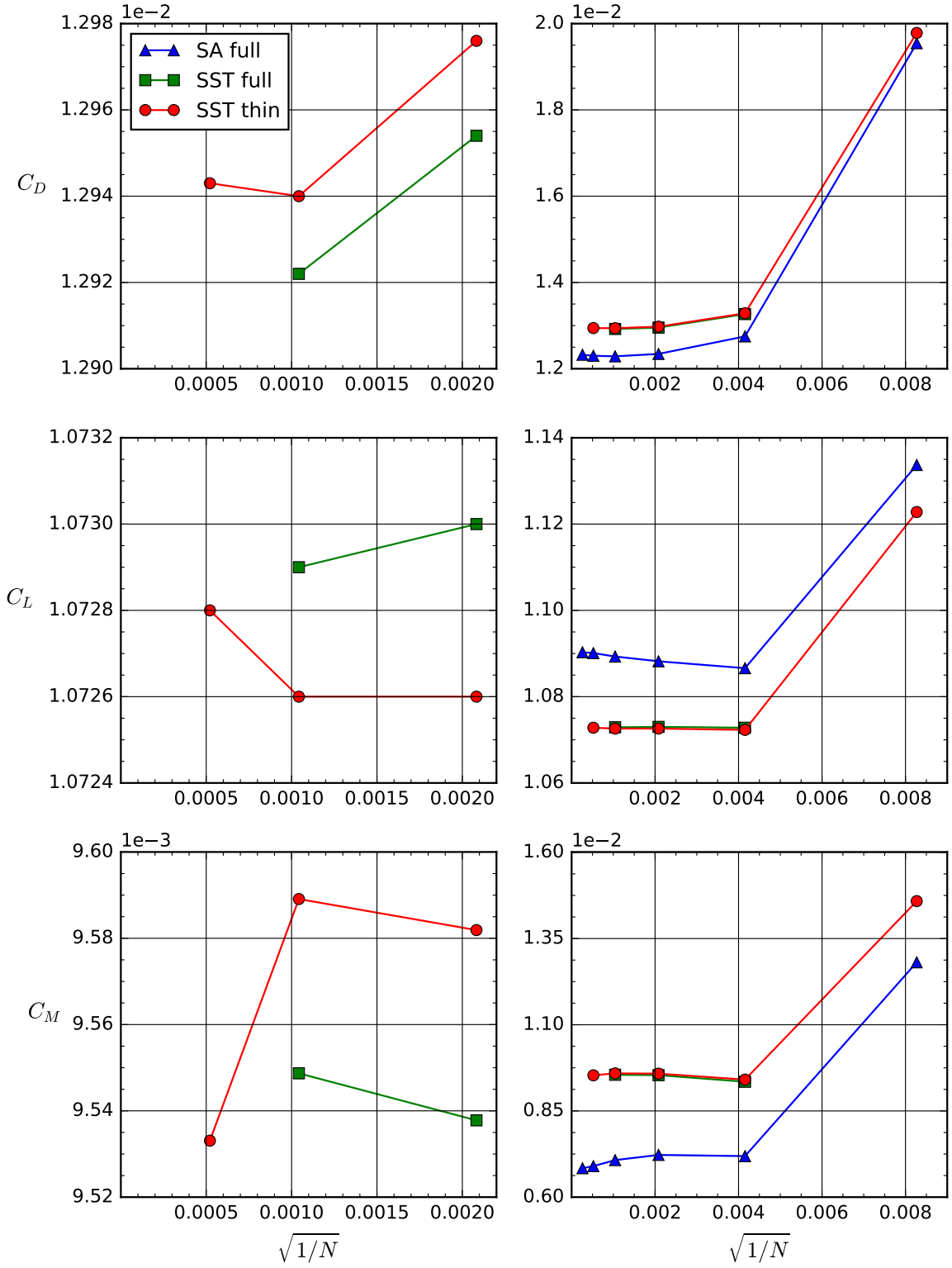

Figure 5.7 Grid convergence of aerodynamic coefficients with SST.

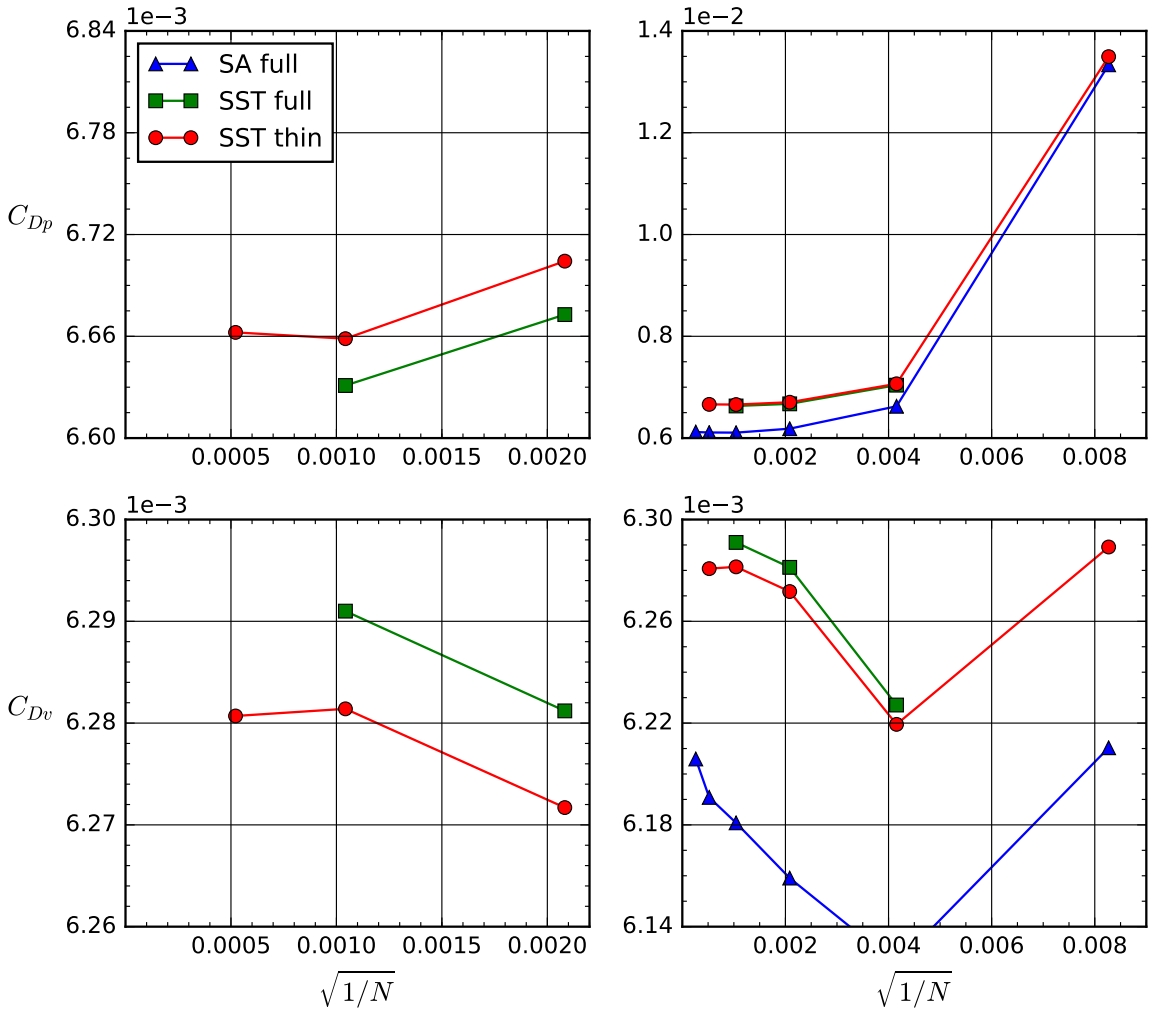

Figure 5.8 Grid convergence of pressure and viscous drag with SST.

The SST model leads to the same central observations about the non asymptotic behavior of the single precision version at high grid densities and the effects of the thin layer approximation. However, the coefficients obtained are quite different from the previous ones, highlighting the modeling uncertainty in turbulence closures.

The results with both models show that the level six grid is too coarse to model the flow. At level five the grid provides a sufficient setting to discretize the governing equations, including the equations of turbulence variables. Thus, the level five grid will suffice in practice and is used in the validation study here. Also, levels one to five can be used to estimate the order of convergence and the discretization error. This is complicated by the different approaches to a limit by the pressure and viscous forces, as demonstrated by the C_{Dp} and the C_{Dv}. Thus, the main coefficients C_D, C_L, and C_M do not show a clear order of convergence, due to them being sums of their pressure and viscous parts. The coefficients C_{Dp} and C_{Dv}, however, could be scaled to a horizontal axis of $\sqrt{1/N^p} \propto \Delta x^p$ to find the proper order p where the curves become linear. The scaled curves are drawn in Figs. 5.9 and 5.10.

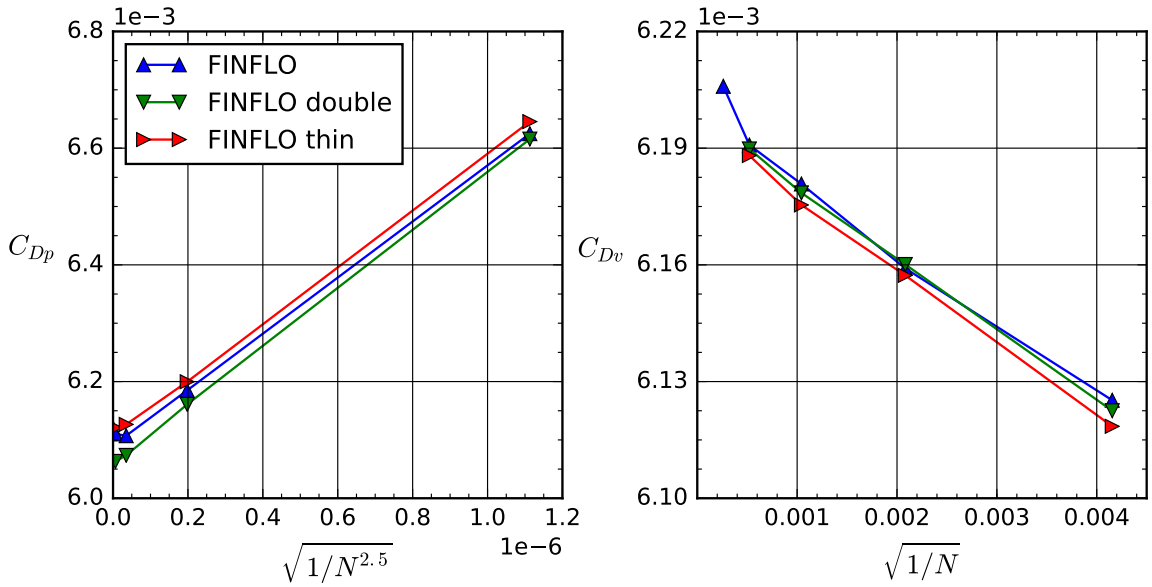

Figure 5.9 The orders of convergence for pressure and viscous drags with SA.

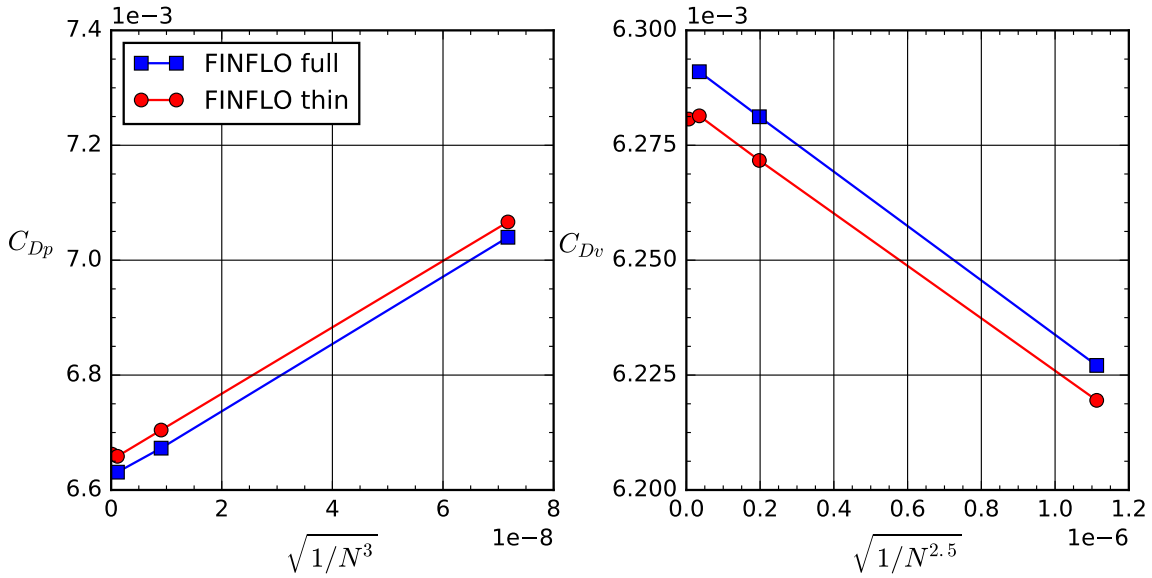

Figure 5.10 The orders of convergence for pressure and viscous drags with SST.

The double precision version acts predictably down to level two, and would presumably continue this at level one. The single precision version breaks the pattern at the two lowest grid levels, with the exception of C_{Dv} with the SA model, where it only breaks at level one. With these exceptions, the orders of convergence were successfully determined and are listed in table 5.2. The variable precision did not affect the order at grid levels three to five.

Table 5.2 The orders of convergence p for C_{Dp} and C_{Dv}.

Model	$p(C_{Dp})$	$p(C_{Dv})$
SA	2.5	1
SST	3	2.5

In this regard the SST model performs much better compared to the SA model, which results in $p(C_{Dv}) = 1$. Note that this is also where the results given by FINFLO differ the most from those give by CFL3D, which converges C_{Dv} much faster, meaning that the $p(C_{Dv})$ obtained here is likely a poor representation of what the SA model is capable of. Nevertheless, the SST model realizing the theoretical order of $p = 3$ for the pressure force and reaching close it in the viscous force is impressive.

The estimated fractional discretization errors E_{level} calculated with the Richardson extrapolation (3.11) for each grid level are presented in the table 5.3. The solution method of friction terms had little effect on these, because the difference of coefficients computed with and without full terms is nearly constant at all levels. Therefore, the E_{level} was calculated from the series with most levels solved: Full terms on the SA model and the thin layer approximation on the SST model.

Table 5.3 The estimated fractional discretization errors of C_{Dp} and C_{Dv}.

Series	Coef.	E_1	E_2	E_3	E_4
SA	C_{Dp}	-0.031%	-0.011%	0.27%	1.5%
single	C_{Dv}	-0.24%	-0.16%	-0.35%	-0.55%
SA	C_{Dp}		0.036%	0.31%	1.6%
double	C_{Dv}		-0.18%	-0.30%	-0.61%
SST	C_{Dp}		-0.0079%	0.098%	0.77%
single	C_{Dv}		0.0024%	-0.033%	-0.18%

Again, the single precision version only provides reasonable E_3 and E_4 due to its behavior at levels one and two, while the double precision version also gives a sensible E_2. These could perhaps be used to estimate the error of the total C_D as

$$E_{level}(C_D) = \frac{C_{Dp}}{C_D} E_{level}(C_{Dp}) + \frac{C_{Dv}}{C_D} E_{level}(C_{Dv}) \tag{5.1}$$

for a single calculation, but for the purpose of evaluating the error in the code, we may simply take $E_{level}(C_{Dp})$ and $E_{level}(C_{Dv})$ for pressure and viscous forces, respectively.

5.4 Skin friction and pressure

The x-wise skin friction coefficient C_{fx} for both upper and lower airfoil surfaces are plotted in Fig. 5.11 to examine the effects of machine precision and grid density. The results are from grid levels two and four, solved with both single and double precision versions of FINFLO using the SA model. The figure includes a zoom on the trailing edge. The comparable curves of the pressure coefficient C_p at the wall are split into Figs. 5.12 and 5.13.

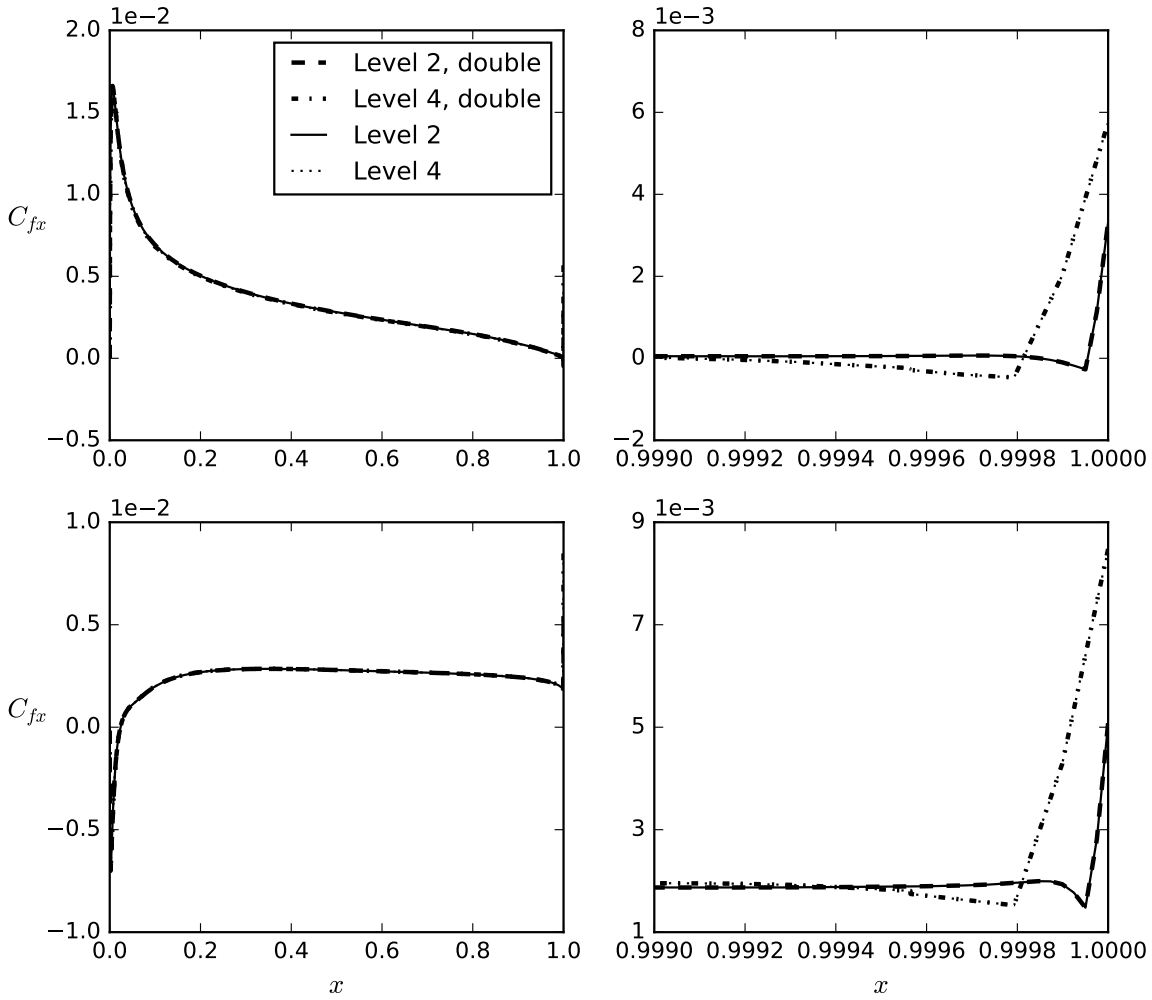

Figure 5.11 The x-wise skin friction, FINFLO, SA. Upper surface on the top and lower on the bottom row. Double marks the double precision version.

The machine precision makes no visible difference at this scale, but the importance of grid refinement at the leading edge is clarified. The geometric near-singularity scrambles the velocities at the last two cells, and the lengths of these cells determine the error magnitude, explaining why the grid family II provided the best grid convergence.

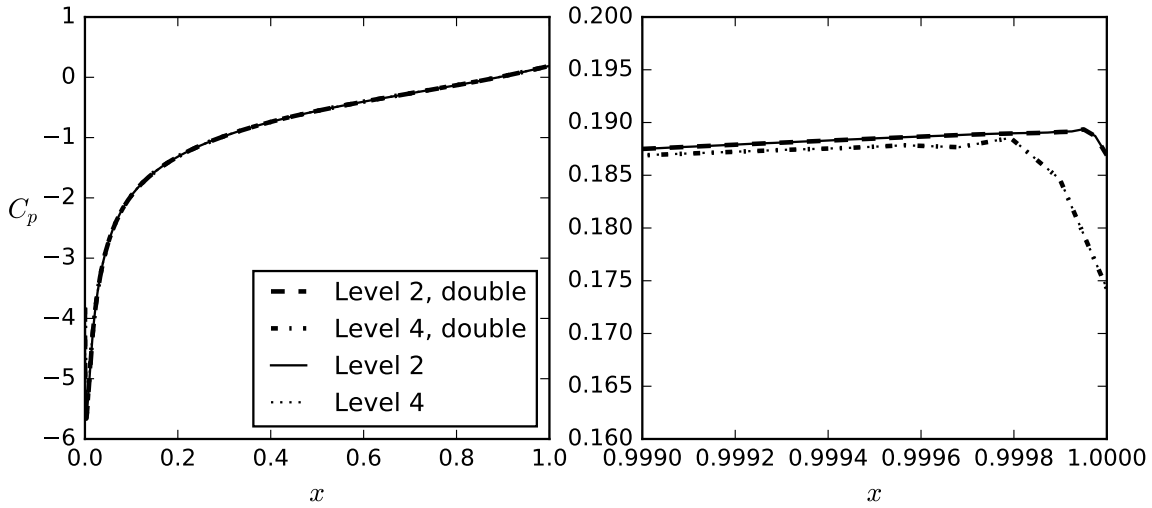

Figure 5.12 Pressure on the upper surface, FINFLO, SA.

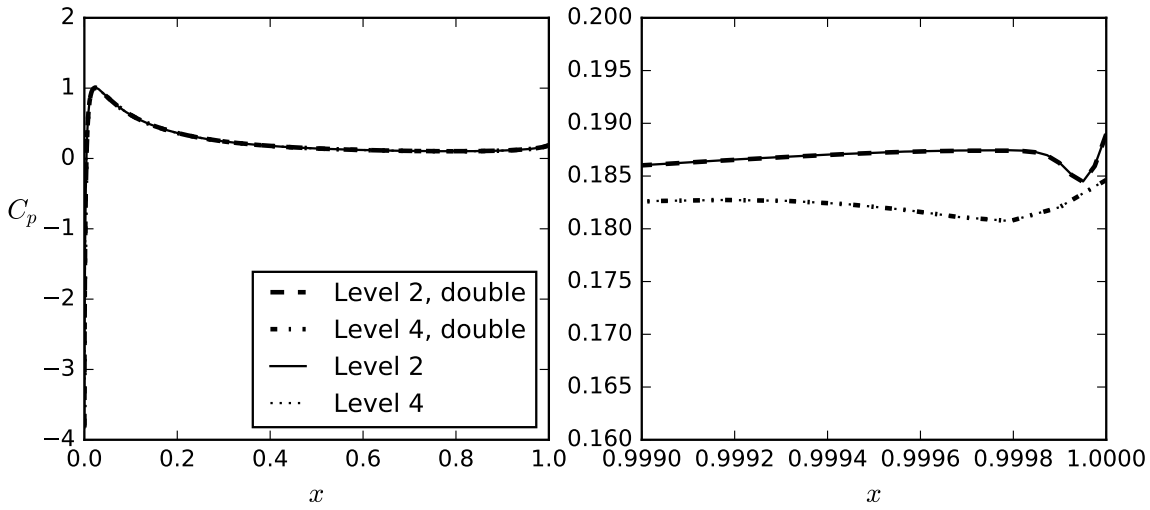

Figure 5.13 Pressure on the lower surface, FINFLO, SA.

The pressure shows a behavior similar to friction. The effects of the near-singularity are less dramatic, but reach farther along the lower surface.

The skin friction and pressure distributions at the airfoil edges on the grid level one are shown in Fig. 5.14 for the benchmarked codes and in Fig. 5.15 for FINFLO.

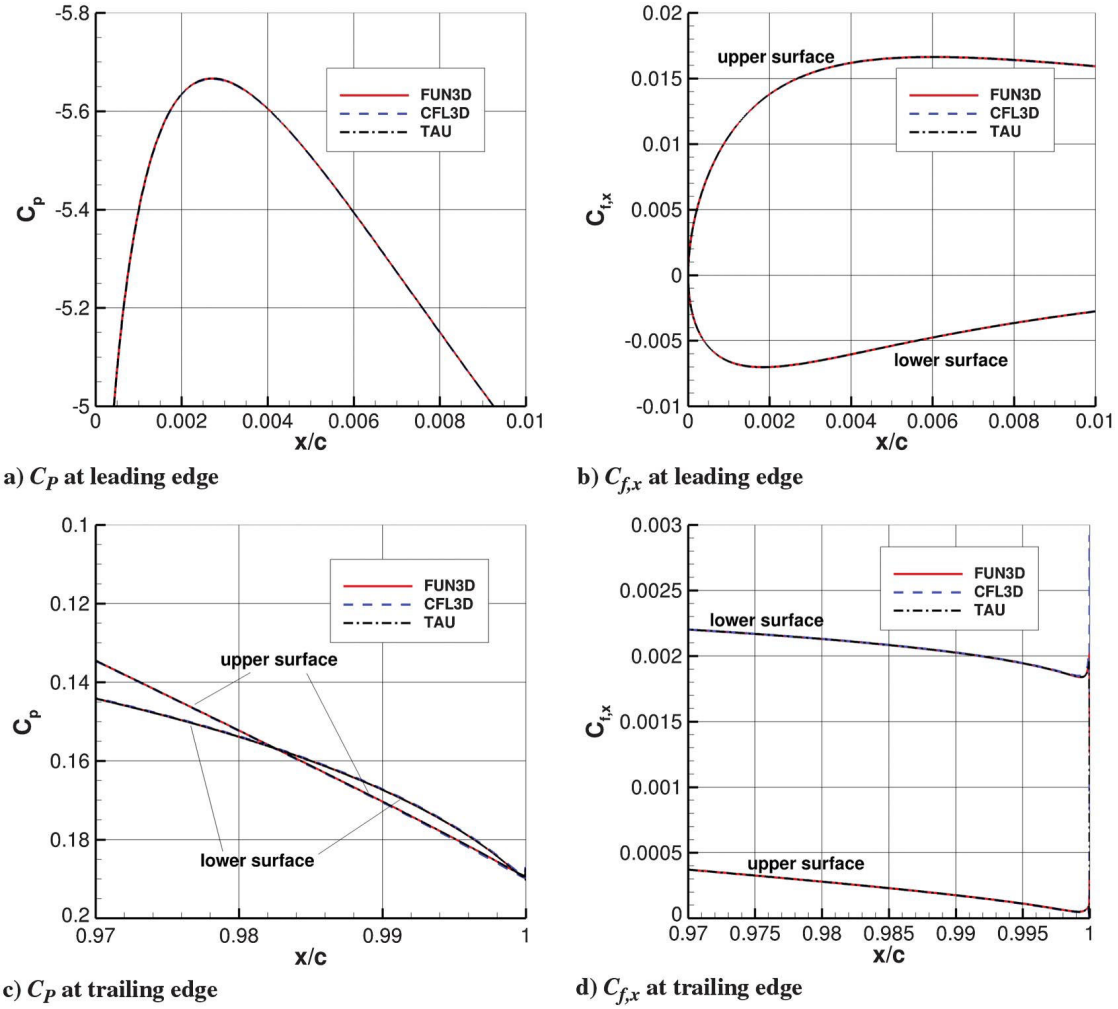

Figure 5.14 Benchmark skin friction and pressure near the edges (Diskin et al., 2016).

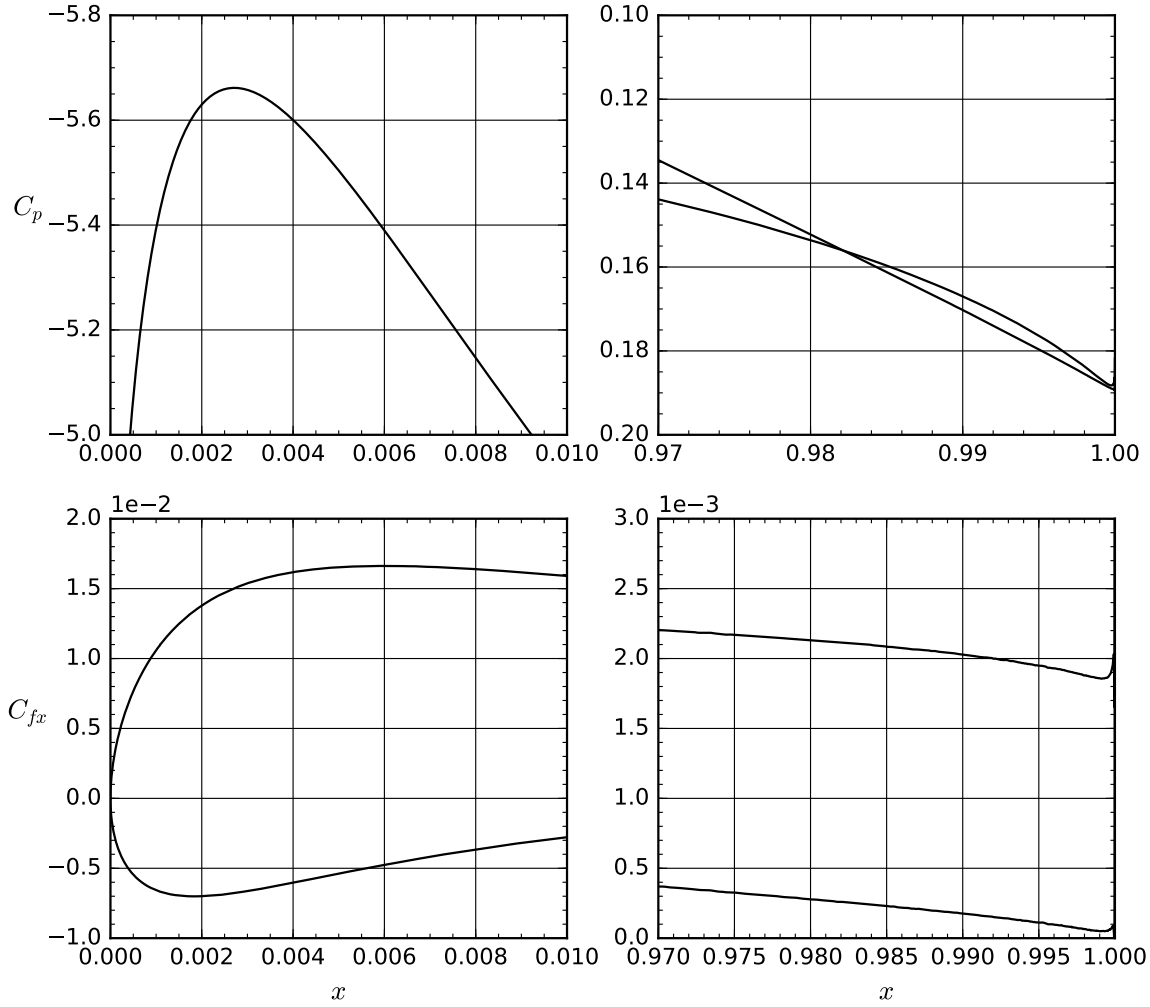

Figure 5.15 Skin friction and pressure computed by FINFLO, SA, near the edges.

On this scale the four codes do not differ from each other in the solutions of the skin friction and pressure. Higher zooms of the benchmark results on the points of maximum C_p and C_{fx} and on the trailing edge can be found in Fig. 5.17. The results given by FINFLO are drawn for comparison in Figs. 5.16 and 5.18.

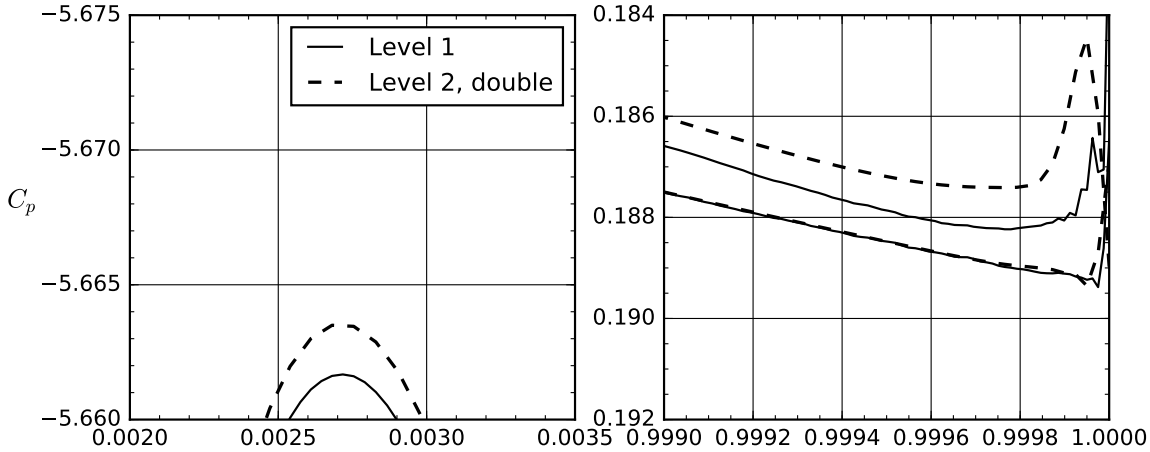

Figure 5.16 Surface pressure computed by FINFLO, SA, high zoom.

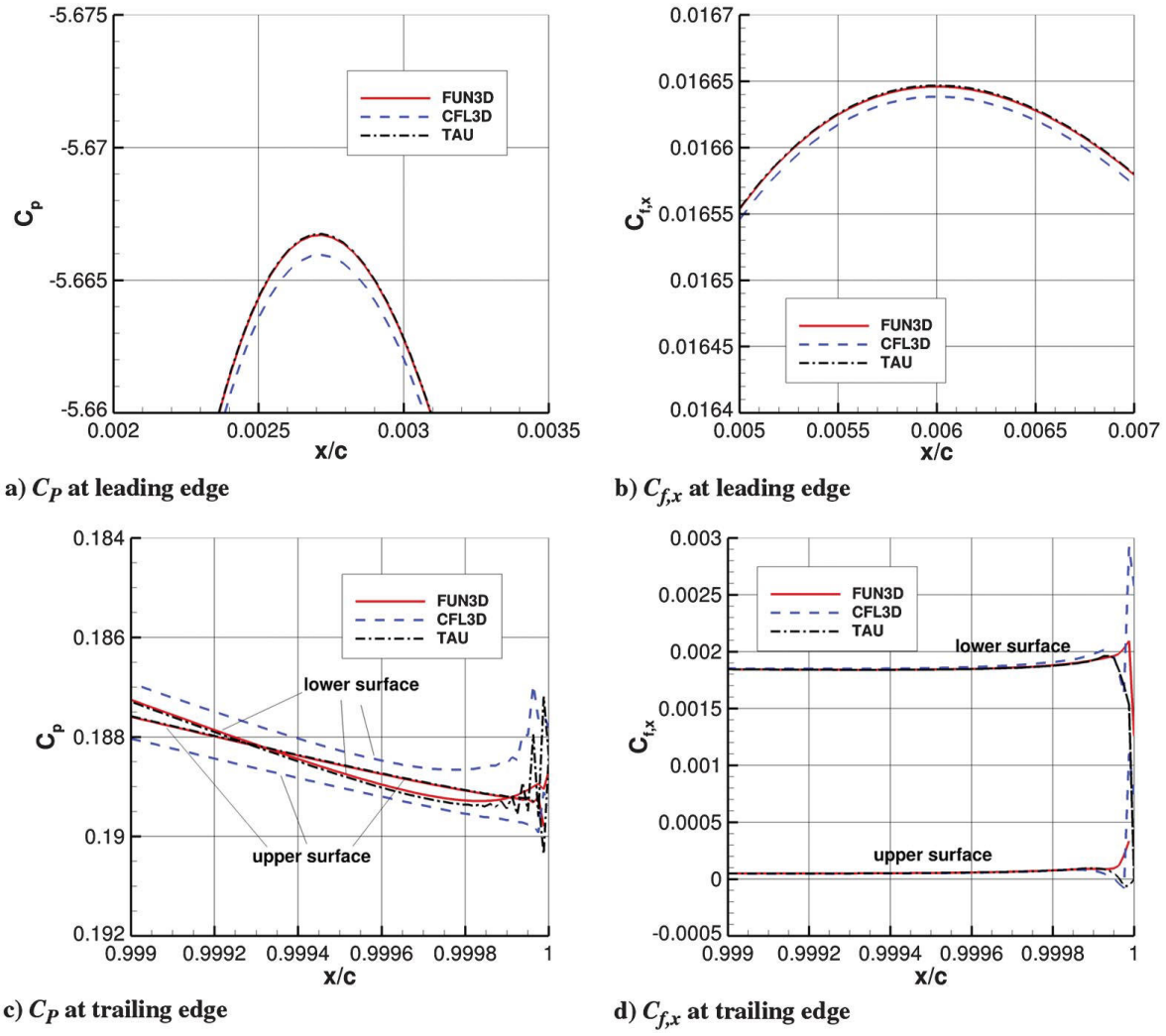

Figure 5.17 Benchmark skin friction and pressure, high zoom (Diskin et al., 2016).

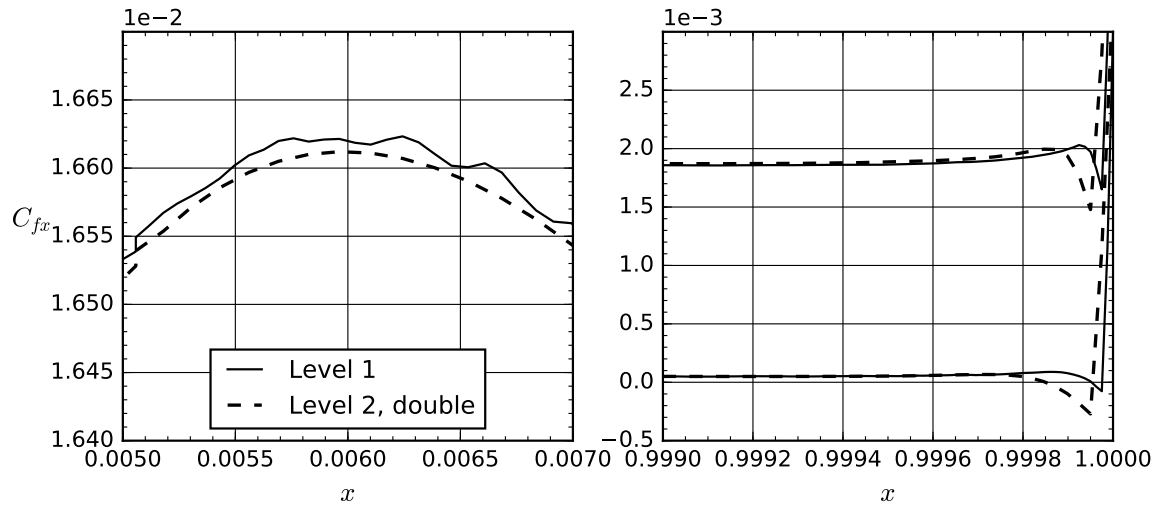

Figure 5.18 Skin friction computed by FINFLO, SA, high zoom.

The results of double precision FINFLO closely follow the same patterns as the results of CFL3D, and the differences likely stem from these being solutions on different grid

levels. The single precision version produces slightly oscillating solutions for both pressure and friction, where the pressure near the trailing edge is especially unstable. This suggests that the 32 bit variables are not long enough to accurately describe the thermodynamic state of the fluid.

The TAU code also shows a wavy pressure curve here. This is probably a manifestation of numerical dispersion caused by the central differencing scheme employed, as the dispersion is prone to creating such regular patterns. Apparently the added artificial viscosity is not quite enough around the geometric near-singularity, although it still makes no difference in the overall results. Another interesting observation is that FUN3D is the only code which does not give a negative skin friction, indicating flow detachment, at the trailing edge of the upper surface.

5.5 Convergence histories

The SA and SST models did not differ much in the convergence to a steady state on a grid level. The machine precision, however, had notable effects in this regard. The lift coefficients C_L as functions of the computational cycle, given by the SA model on both machine precisions, are presented in Fig. 5.19. Similar plots for the residuals of x-wise momentum and turbulent viscosity are in Fig. 5.20.

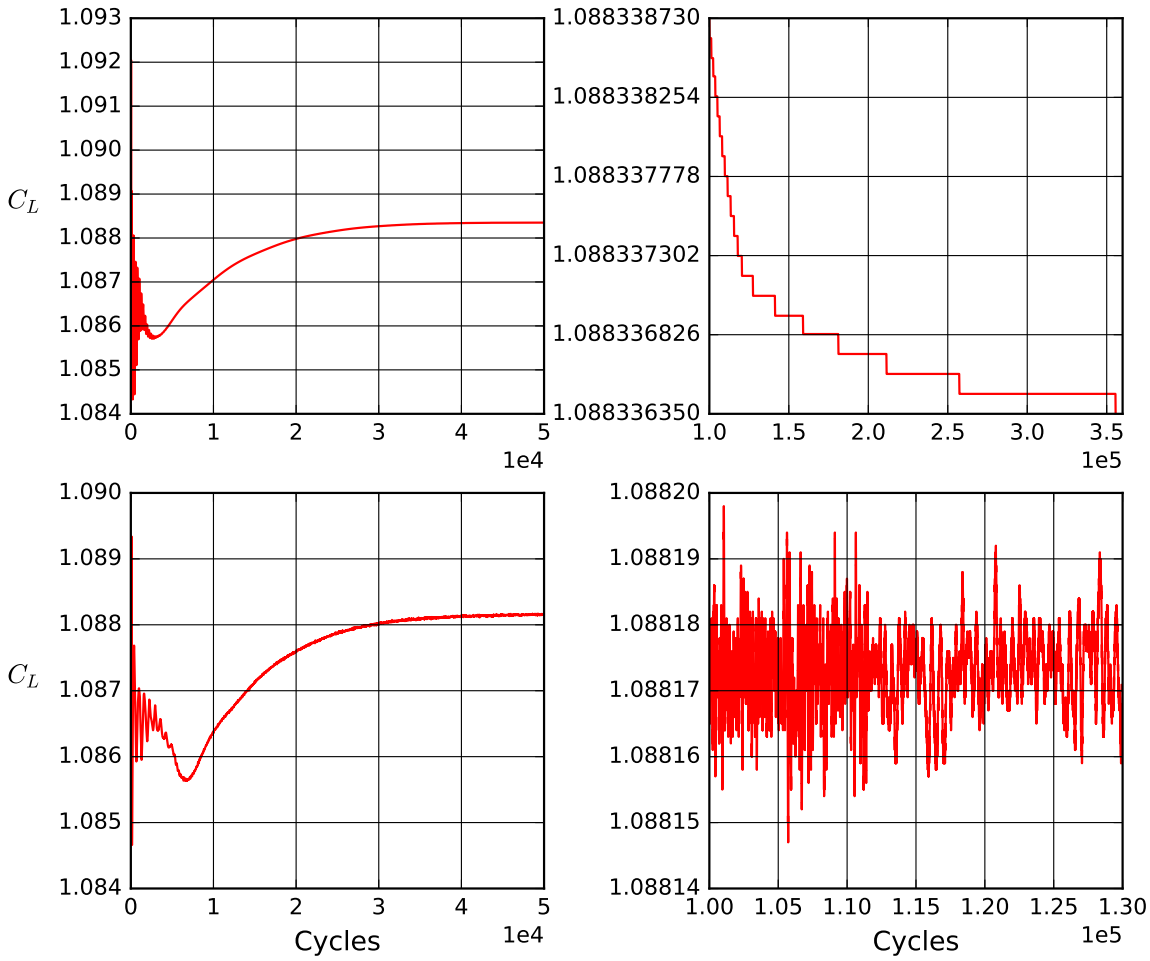

Figure 5.19 The development of lift with the SA model. Double precision FINFLO on the top row and single precision on the bottom.

Both versions converge the lift to five significant digits in about 50 000 cycles with a Courant number of 1.9. The results differ a little, as the double precision gives $C_L = 1.0883$ and the single $C_L = 1.0882$. After 50 000 cycles the double precision version keeps adding significant digits while the other version oscillates around the same average $C_L \approx 1.088175$. With the drag the relative amplitude of the oscillation was stronger and such averaging had to be used to determine enough digits for the purpose of plotting the grid convergence. This is due to the fluctuation of viscous forces being higher than that of pressure forces, which could be clearly seen in the histories of C_{Dv} and C_{Dp}, for brevity not included here, but also visible in the figures of the previous section.

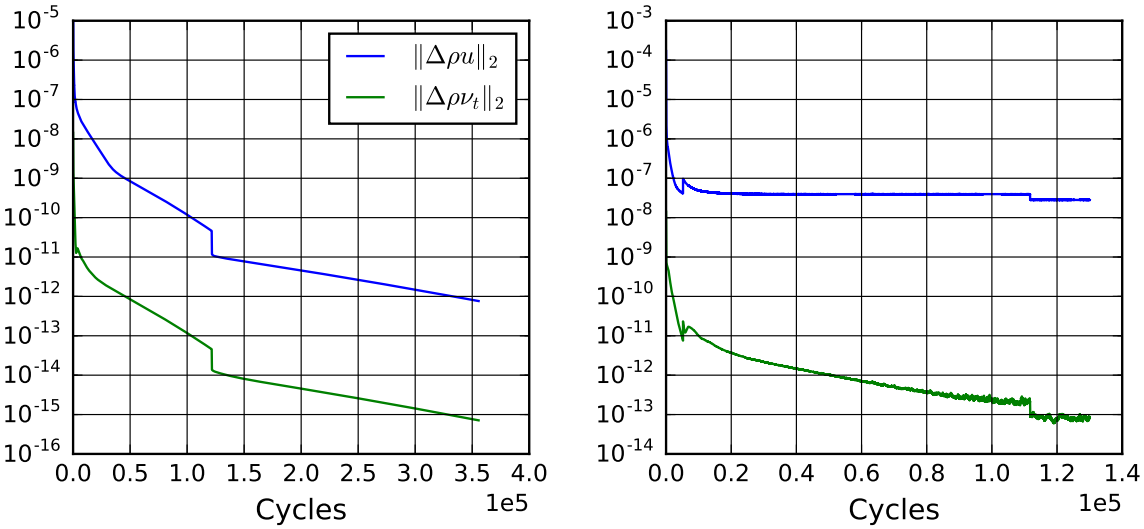

Figure 5.20 L2-norms of x-wise momentum and turbulent viscosity residuals with the SA model. Double precision version on the left and single on the right.

The double precision simulation was stopped when the density residual hit 10^{-12}, and the single precision after the the residuals of all quantities reached machine convergence. The turbulent viscosity was the last one to do so. The discontinuities are caused by changing the number of levels used by the multigrid algorithm. The simulations started with two levels to accelerate solution, and later switched to one level to avoid any error caused by the algorithm. The single precision simulation was accidentally started with one level, which is why there is a jump upwards in the residual histories at a couple of thousand cycles. Turning off the multigrid acceleration slows the rate at which C_L changes and makes the solution given by the single precision version less restless in the frequency of its oscillation, but only slightly reduces its amplitude.

5.6 Flow profiles

This section presents some boundary layer profiles and color-coded pictures of the flow around the airfoil. Fig. 5.21 has the simulated u^+ plotted against y^+, similar to the measured values shown as an example in Fig. 2.4, plus the ratio of eddy to molecular viscosity ν_t/ν on the same scale. The profiles are taken from the grid levels one and five, as the level five was the coarsest at which the boundary layer was formed correctly. The code precisions and turbulence models are also compared. Full friction terms were calculated in these simulations.

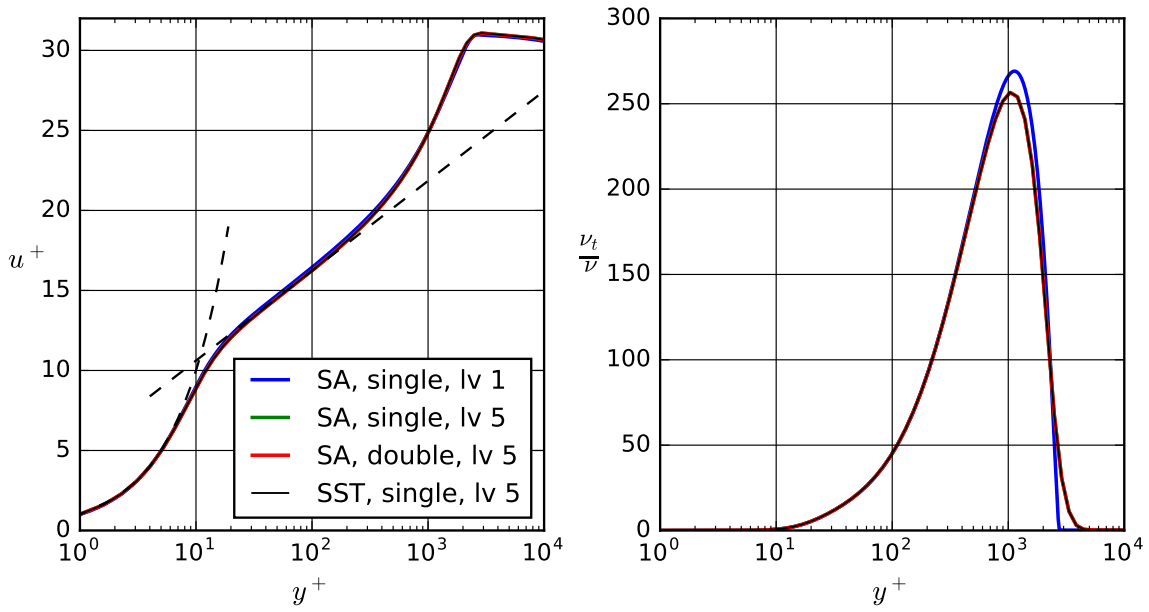

Figure 5.21 Boundary layer profiles of u^+ and ν_t/ν. Dashed lines show the linear and logarithmic laws.

The profiles are from the upper surface at $x = 0.304074$, as this is the point at grid level five closest to $x = 0.297174$, where the modified NACA 0012 surface geometry has $\mathrm{d}y/\mathrm{d}x = 0$ and thus the y-axis is normal to the surface. This makes y^+ less ambiguous. All of the nondimensional velocity profiles are essentially indistinguishable. The actual velocity profiles must then differ a little, due to the different grid levels, models, and machine precisions producing different wall frictions, with which the u^+ is nondimensionalized. The models and machine precision also produce the same ν_t/ν profiles. The denser grid level gives a bit higher maximum ν_t/ν and steeper descent for it in the upper layer, but makes no difference in the logarithmic layer, which here extends to $y^+ \approx 300$.

Fig. 5.22 consists of two color-coded pictures depicting the pressure coefficent and eddy viscosity near the airfoil. These show how the airfoil affects the pressure and turbulence in the flow field around it.

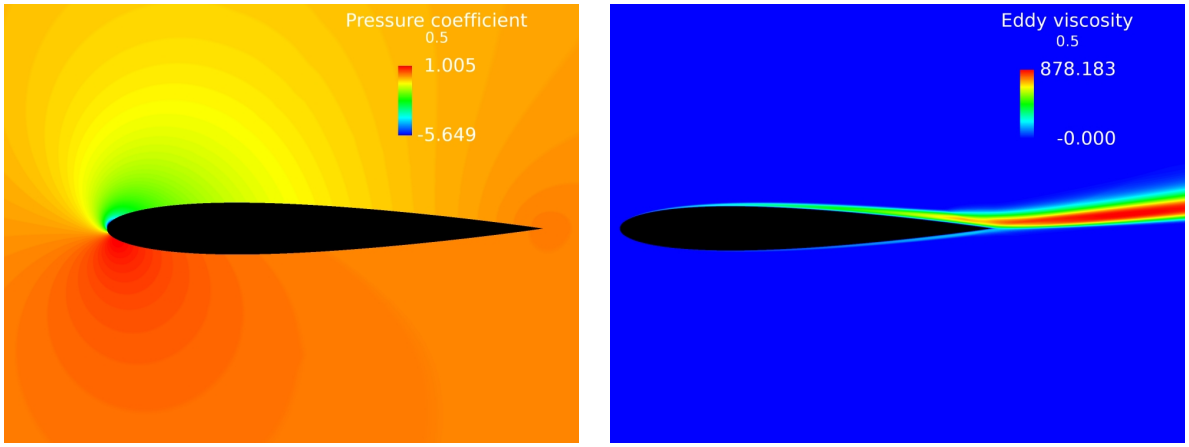

Figure 5.22 Pressure coefficient C_p around the airfoil on the left and the eddy viscosity ν_t on the right. SA model, grid level two, single precision FINFLO.

These kind of pictures are a little difficult to analyze, but they do help show that the code and the model behave reasonably. The left picture shows the stagnation point at the lower surface of the leading edge and the region of underpressure above the airfoil. The right picture visualizes how the SA model, and non-algebraic RANS models in general, transport and disperse turbulence as regions of increased mixing.

5.7 Computing time

Compiling the double precision version raised the question about its effects on the computing time, with the most pessimistic view being that the time would double and the most optimistic that there would not be much of a difference on the modern 64-bit processors. To curb speculation with experimentation, both versions were set to run the first 1000 cycles of the benchmark flow on grid level four with some different parameters and no multigrid acceleration. The runs were repeated with an increasing number of cores, up to 64 where possible. The double precision version refused to run on more than 16 cores, and no fix could be found in time for this work. Thus, the test is quite limited in scope but revealed the general trends of the computation time.

To revise the hardware: Desktop Intel Sandy Bridge (SNB) E5-1650 3.20 GHz, CSC Sandy Brigde Intel E5-2670 2.60 GHz, CSC Haswell (HSW) Intel E5-2690v3 2.60 GHz. The used processor time per core is plotted in Fig. 5.23. This shows how different options affect the time used and how parallel processing increases the time billed by a service provider, such as CSC, if one is used. For example, if time per core on eight cores is twice that of the time per core on a single core, solving the problem on eight cores is twice as expensive as it would be on a single core. The times taken per cell each cycle on a single core using single precision, SA model, and full friction were as follows: Desktop 4.27 µs, CSC SNB 4.38 µs, and CSC HSW 3.90 µs. Using double precision, the times were: Desktop 6.17 µs, CSC SNB 6.31 µs, and CSC HSW 5.19 µs.

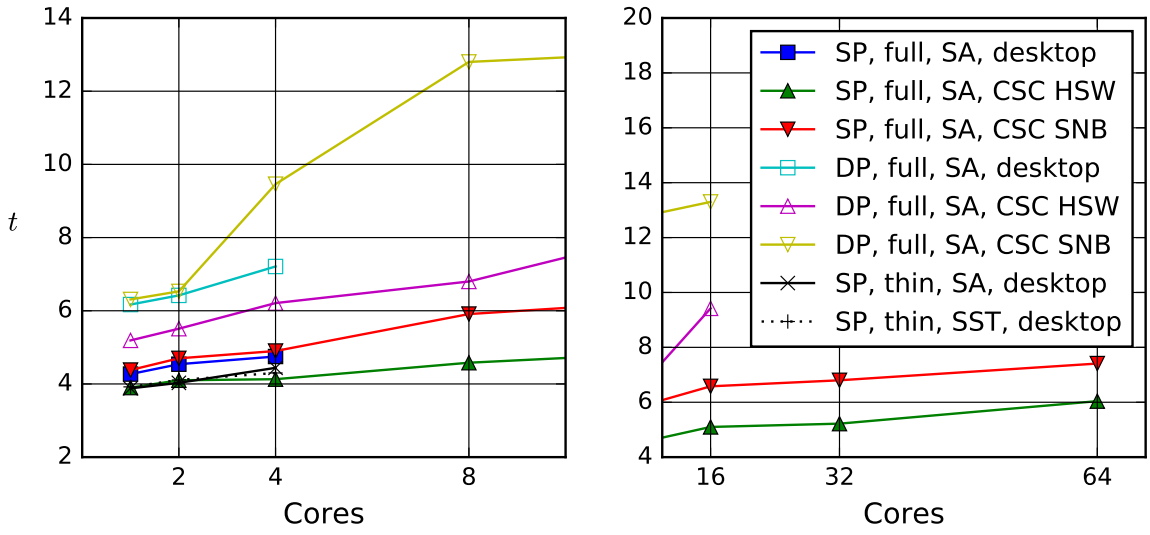

Figure 5.23 Processor time per cycle, cell, and core in microseconds. SP and DP denote single and double precisions, full and thin the friction solution accuracy.

The newer HSW processor is, as could be expected, faster than the two older SNB processors in 32-bit precision operations, and the difference is even clearer in 64-bit

precision. On a single core, doubling the precision increases the computing time by nearly 50% on the SNB, while on HSW the increase is closer to 30%. The reported base clock frequencies do not seem to be very descriptive of the computing power.

The calculations with a thin layer approximation seem to take about 10% less time than those with full friction terms. Unexpectedly the the SA and SST models are no different in this regard, despite the SST model having one more equation to solve. One of the possible reasons is the SA model function f_w (4.17), which includes a sixth root, probably being a heavy iterative operation on a computer.

Fig. 5.24 displays how the computing power scales in parallel processing. The black line shows the ideal situation where the power scales linearly with the number of cores.

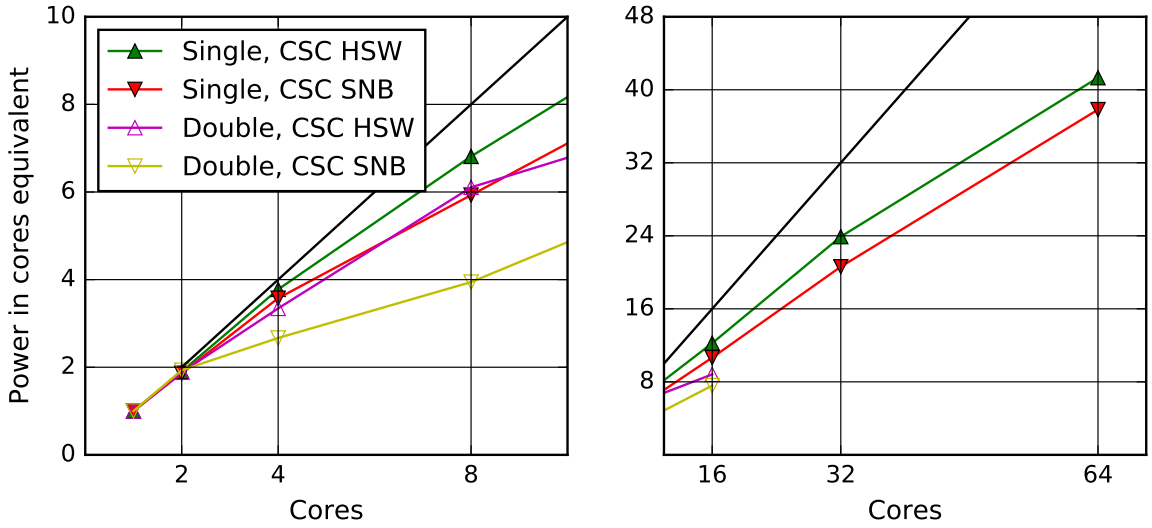

Figure 5.24 Parallel computational power in relation to the ideal situtation where doubling the number of cores would halve the time taken.

The relatively good single core 64-bit performance of the HSW does not extend well to parallel processing, and the double precision computing power scales poorly with the number of cores compared to the parallel single precision calculations. On 16 cores, the HSW solves the single precision operations 12 times as fast as it would on a single core, while with double precision it is only about nine times as fast. The amount of information exchanged in between cores apparently becomes a problem, especially so for the CSC SNB processor, which produced quite a peculiar graph. This series was thought to have suffered from some temporary problem and was repeated, only to receive the same times again.

5.8 Angle of attack

This section discusses the validation study, where the simulated coefficients are compared to those measured in a wind tunnel experiment of the NACA 0012 airfoil at different angles of attack α (Ladson, 1988). The experimental study is quite exstensive, consisting of measurements with different Mach and Reynolds numbers, of which the measured series closest to the benchmark flow was chosen for this validation. In addition to tests with free turbulent transition, the report includes tests with fixed tran-

sition by a strip of abrasive attached on the airfoil leading edge. These fixed transition measurements are specifically intended for a comparison with simulations employing turbulence models unable to predict transition, such as the two models applied here. The test were repeated with abrasives of different grit sizes, and the size here is chosen so that the measured lift at $\alpha \approx 10°$ is the closest to the lift simulated in the verification study. The experimental series used is defined as $Ma = 0.15$, $Re_L = 5.95 \cdot 10^6$, fixed transition No. 80 grit (Ladson, 1988).

Thus, the simulations discussed in this section differ from the benchmark flow slightly in Re_L and more in α, but have otherwise the same boundary conditions. Two series were computed, one with the SA and one with the SST model, both using the single precision FINFLO. Based on the verification results, the grid level four was deemed the best compromise of computation time and accuracy. A validation would ideally use a more realistic blunt-ended airfoil geometry, but for consistency the sharp-edged grid used in verification is also employed here. The experimental series contains measurements at 17 different α, but for the purpose of this brief validation, only seven of these, listed in table 5.4, were simulated. These include the most important points of interest, the point of highest lift per drag at α_4 and the points α_6 and α_7 in between which flow detachment happens. The results are plotted in Fig. 5.25 and the fractional difference $\Delta C/C_e = (C_{simulation} - C_{experiment})/C_{experiment}$ in Fig. 5.26.

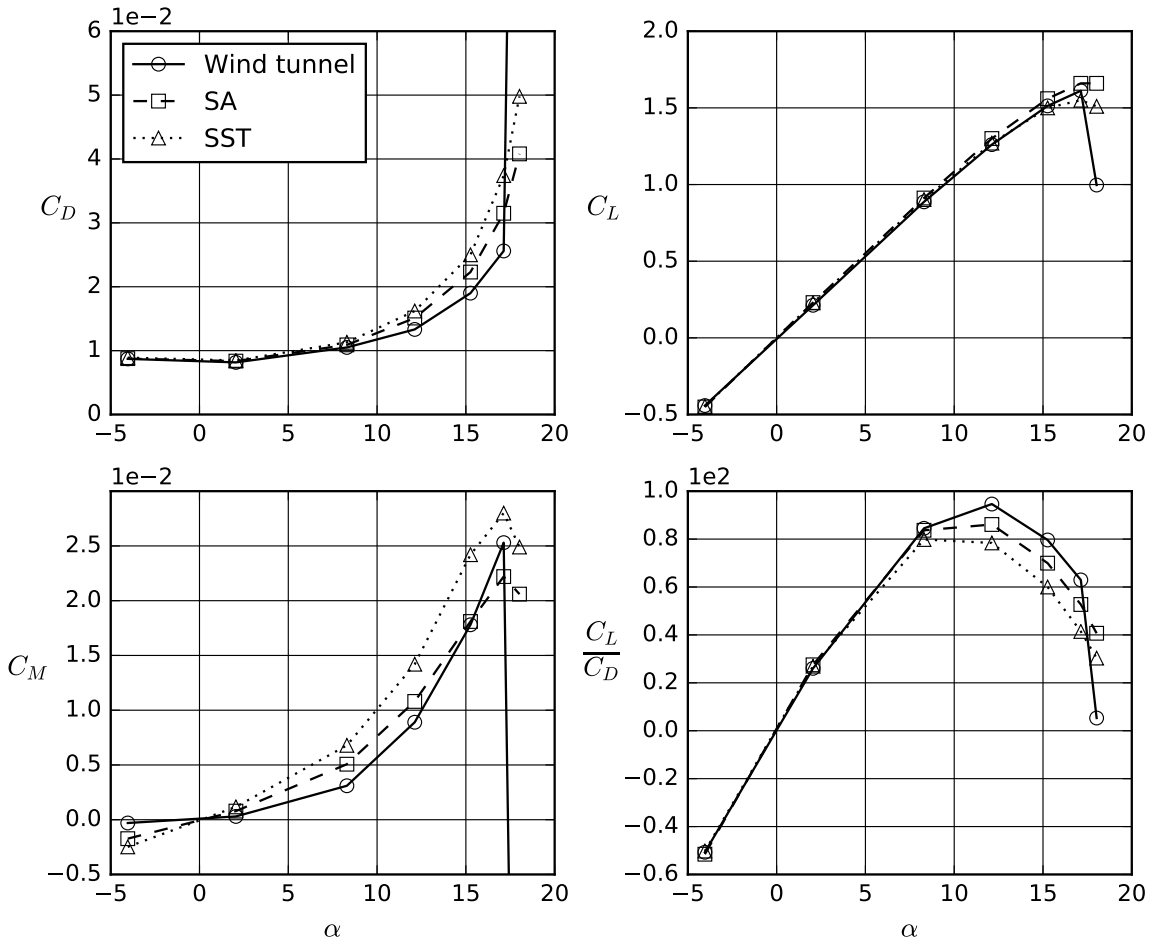

Figure 5.25 The main aerodynamic coefficients, plus the ratio of lift and drag, as functions of the angle of attack α. Results given by the single precision FINFLO using the SA and SST models are plotted against wind tunnel measurements (Ladson, 1988).

Table 5.4 The simulated angles α.

Measurement	1	2	3	4	5	6	7
α	-4.04	2.05	8.30	12.12	15.26	17.13	18.02

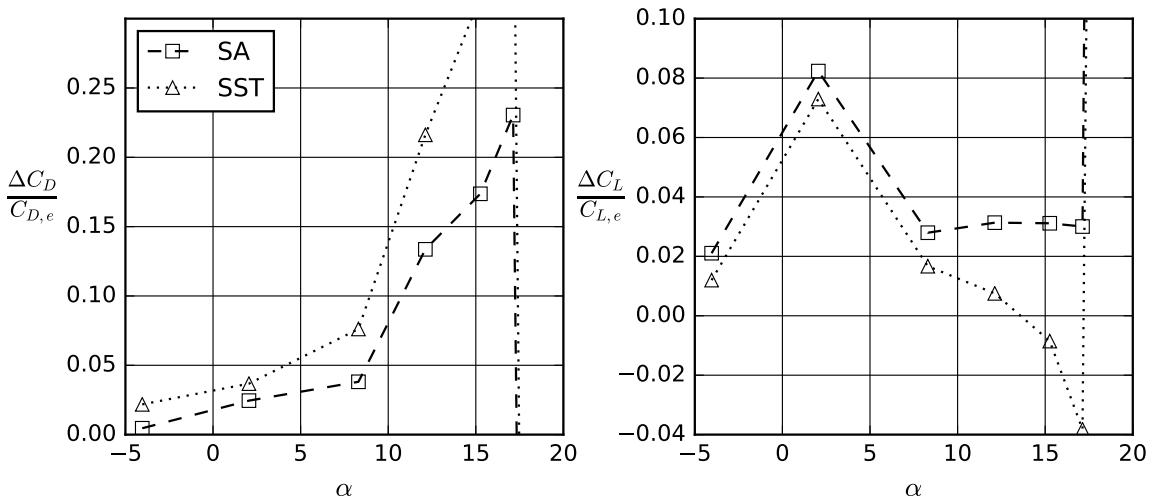

Figure 5.26 The fractional difference of simulated and experimental results.

Both models predict the lift C_L quite well, with the difference to measurements only being a few percent up until the point of flow detachment, with the exception of $\alpha \approx 0°$, where also $C_L \approx 0$ and the smallest error results in a large fractional difference. The drag C_D, however, is systematically overestimated at $\alpha \geq 5°$. The pitching moment C_M is also quite a bit off, but this probably due to its point of measurement $x = L/4$ being chosen to yield a very low C_M, making it difficult to predict.

The results display the limits of linear RANS modeling, also mentioned in chapter 2. As the angle of attack increases, the pressure and density above the airfoil decrease, and the assumption of symmetric fluid dilation included in the Boussinesq approximation (2.21) fails. A nonlinear Reynolds stress scheme would likely improve the results at higher α, as such schemes have generally showed improvement over linear modeling in cases of notable density differences (NASA, 2017c). Nevertheless, as the point of flow detachment approaches, the scale of turbulence grows and the approximation that the turbulent eddies only act as increased mixing becomes less accurate. This is most evident at α_7, where the flow in the wind tunnel has detached but the simulated flow still mostly sticks to the airfoil. Accurate simulation near and above the point of separation would certainly require the application of DES instead of pure RANS modeling.

It is somewhat surprising that despite not properly simulating the flow detachment here, both RANS models are able to accurately predict the point of detachment by showing falling C_L and C_M after it.

Chapter 6

Conclusions

Despite the difficulties faced, the primary objective of this work was met. This was to demonstrate that FINFLO converges to the same analytical solution as other codes when the discretization error approaches zero. This end result is beneficial to the credibility of all four codes included in the comparison, as it is indeed very unlikely that four faulty implementations of numerical methods would approach the same solution by chance. Reaching for the primary objective led to fixing the implementation of the SA model in FINFLO and to compiling a double precision version of the code. The former outcome is simply a useful result of the verification process, while the latter has more complex implications.

The observation that the double precision version of FINFLO behaves similar to other double precision codes proves that FINFLO approximates the RANS and SA model equations correctly. The very same verified numerical implementation of the governing equations is applied in the single precision version, which, therefore, can also be considered verified, as its deviance from the benchmark results was shown to stem from the computer round-off error. The magnitude of this error was less than a percent on all but the two densest grid levels, where it begins to grow. On the grid level three the machine error is comparable to the discretization error and on the levels four and five well below it.

The double precision version is more accurate and able to provide more than three significant digits. It also makes the results easier to read, or post-process, due to the non-oscillating solution. However, in practical applications, where the grids used are not nearly as dense as the level two here, the single precision machine error would be insignificant compared to the discretization and especially the modeling errors. These would make it difficult to justify taking more than three significant digits anyway.

Nowadays the computer memory required for double precision computing is a non issue. For the simulations conducted here, the memory requirements were nowhere near the capacity provided by CSC. Even the desktop computer with a 16 GB memory was able to run the double precision simulations at grid level three. Nonetheless, the variable size had a clear and undesirable effect on the processor time taken, most notably on how the computing power scaled with the number of cores used.

Doubling the number of bits which the cores had to pass to each other made parallel processing much less efficient, revealing a kind of a bottleneck in this regard. The newer Haswell processor architecture was less slowed down by the double precision on a single core compared to the older Sandy Bridge, and it also scaled better in parallel processing, demonstrating some technological advancements in 64-bit variable operations. Nevertheless, even with Haswell the scaling was still much better in single

precision, and at 16 cores doubling the variable precision also doubled the time taken per cycle. Consider this with the finding that the variable precision had little effect on the number of cycles needed to reach a converged solution, and the disadvantage of double precision computing is evident. Employing the single precision version in practical simulations would leave much more processor time for other tasks, such as a grid refinement study for estimating the discretization error in the simulation. The proper allocation of the processor time available is probably never an easy task, and these findings unfortunately add yet another question to be answered.

The validation study illustrated the shortcomings of linear turbulence modeling, in this case limiting their use to low angles of attack. The poor predictions of drag at the higher angles are more likely to be caused by the Boussinesq approximation rather than the models themselves, as both of them have shown better results when coupled with nonlinear turbulent stress schemes. On a more optimistic note, even with the linear scheme both models estimated lift quite well up to the point of separation. Also, the accuracy with which the points of separation and highest lift per drag were predicted was remarkable.

The SA model equation responded unexpectedly to the different discretizations of its convection term, with a higher order discretization producing a larger error. This does not affect the outcome of the verification, as all discretizations approached the same limit, but it does bring about a possible future improvement in FINFLO. Currently, FINFLO has options to set the discretizations of convection terms separately in each direction but not in each equation. The SA model would apparently benefit from the latter.

Bibliography

Auvinen, M. (2017). "Fluid mechanical energy diagrams". Personal communication.

Buckingham, E. (1914). "On physically similar systems; illustrations of the use of dimensional equations". In: *Physical review* 4.4, p. 345.

Ceze, M.A. and K.J. Fidkowski (2016). "High-order output-based adaptive simulations of turbulent flow in two dimensions". In: *AIAA Journal* 54.9, pp. 2611–2625. DOI: 10.2514/1.J054517.

Chien, K. (1982). "Predictions of channel and boundary-layer flows with a low-Reynolds-number turbulence model". In: *AIAA Journal* 20.1, pp. 33–38.

Coles, D.E. and E.A. Hirst (1968). *Compiled Wieghardt's data*. Visited September 8th 2017. URL: https://www.grc.nasa.gov/www/wind/valid/fpturb/WIEVELS.data.

CSC (2017). *IT center for science*. Visited September 27th 2017. URL: https://research.csc.fi/.

Diskin, B., J.L. Thomas, C.L. Rumsey, and A. Schwöppe (2016). "Grid-Convergence of Reynolds-Averaged Navier–Stokes Solutions for Benchmark Flows in Two Dimensions". In: *AIAA Journal* 54.9, pp. 2563–2588. DOI: 10.2514/1.J054555.

DLR (2017). *TAU software system*. Visited October 1st 2017. URL: http://tau.dlr.de/code-description/.

Favre, A.J. (1965). *The equations of compressible turbulent gases*. Aix-Marseille University.

Givi, P. (2016). "Introduction: Evaluation of RANS Solvers on Benchmark Aerodynamic Flows". In: *AIAA Journal* 54.9, pp. 2561–2562. DOI: 10.2514/1.J054642.

Goodwin, W.M. (2015). "Global climate modeling as applied science". In: *Journal for General Philosophy of Science* 46.2, pp. 339–350.

Guillaume, M et al. (2012). "Swiss/Finnish computational fluid dynamics simulation on the F/A-18". In: *Proceedings of the 28th ICAS Congress*. ICAS Paper 2012-10.6.1. Brisbane, pp. 1–6.

Hellsten, A. (1998). "Some improvements in Menter's $k - \omega$ SST turbulence model". In: *29th AIAA Fluid Dynamics Conference*. AIAA paper 98–2554.

IEEE-754 (2008). "IEEE Standard for Floating-Point Arithmetic". In: DOI: 10.1109/IEEESTD.2008.4610935.

Jantunen, P. (2016). *Turbulenssimallit ja niiden reunaehdot tasolevyllä*. Memorandum CFD/MECHA-38-2016. Aalto University School of Engineering. Unpublished.

Jones, W.P. and B.E. Launder (1972). "The prediction of laminarization with a two-equation model of turbulence". In: *International journal of heat and mass transfer* 15.2, pp. 301–314.

Ladson, C.L. (1988). *Effects of independent variation of Mach and Reynolds numbers on the low-speed aerodynamic characteristics of the NACA 0012 airfoil section*. Technical memorandum 4074. NASA.

Laine, S., T. Siikonen, and P. Kaurinkoski (1992). "Calculation of Transonic Viscous Flow Around a Delta Wing". In: *Proceedings of the 18th ICAS Congress*. ICAS Paper 92-4.2.1. Beijing, pp. 286–295.

Lampinen, M.J. (2010). *Termodynamiikan perusteet*. 5th ed. Helsinki, Finland: Otatieto. 182 pp. ISBN: 978-951-672-368-9.

Lombard, C.K., J. Bardina, E. Venkatapathy, and J. Oliger (1983). "Multi-dimensional formulation of CSCM - An upwind flux difference eigenvector split method for the compressible Navier-Stokes equations". In: *6th Computational Fluid Dynamics Conference*, pp. 649–664.

Menter, F.R. (1994). "Two-equation eddy-viscosity turbulence models for engineering applications". In: *AIAA Journal* 32.8, pp. 1598–1605.

Menter, F.R., M. Kuntz, and R. Langtry (2003). "Ten years of industrial experience with the SST turbulence model". In: *Turbulence, heat and mass transfer* 4.1, pp. 625–632.

MPICH (2017). *Message passing interface implementation*. Visited September 7th 2017. URL: `https://www.mpich.org/`.

NASA (2017a). *CFL3D Navier-Stokes solver*. Visited October 1st 2017. URL: `https://cfl3d.larc.nasa.gov/`.

— (2017b). *FUN3D Navier-Stokes solver*. Visited October 1st 2017. URL: `https://fun3d.larc.nasa.gov/`.

— (2017c). *Turbulence modeling resource*. Visited May 4th 2017. Langley Research Center. URL: `https://turbmodels.larc.nasa.gov`.

Pitkäranta, J. (2007). *TKK:n Laaja Matematiikka 2*. Helsinki, Finland: Edita Prima. 1052 pp. ISBN: 952-91-9159-6.

Popper, K. (1959). *The Logic of Scientific Discovery*. London, UK: Routledge. 513 pp. ISBN: 0-415-27844-9.

Rahman, M. (2016). "Applied turbulence modeling". In: *Course Material, Aalto University School of Engineering*.

Reynolds, O. (1894). "On the Dynamical Theory of Incompressible Viscous Fluids and the Determination of the Criterion". In: *Proceedings of the Royal Society of London* 56, pp. 40–45.

Richardson, L.F. and J.A. Gaunt (1927). "The deferred approach to the limit. Part I. Single lattice. Part II. Interpenetrating lattices". In: *Philosophical Transactions of the Royal Society of London, Series A* 226, pp. 299–361.

Roache, P.J. (1998). *Verification and validation in computational science and engineering*. Albuquerque, USA: Hermosa Publishers. 446 pp. ISBN: 0-913478-08-3.

Roe, P.L. (1981). "Approximate Riemann solvers, parameter vectors, and difference schemes". In: *Journal of computational physics* 43.2, pp. 357–372.

Siikonen, T. (1995). "An Application of Roe's Flux-Difference Splitting for the $k-\epsilon$ Turbulence Model". In: *International Journal for Numerical Methods in Fluids* 21.11, pp. 1017–1039.

— (2013). "Laskennallisen virtausmekaniikan ja lämmönsiirron jatkokurssi". In: *Course Material, Aalto University School of Engineering*.

— (2014a). "Laskennallisen virtausmekaniikan ja lämmönsiirron perusteet". In: *Course Material, Aalto University School of Engineering*.

— (2014b). "Virtaussimulointi". In: *Course Material, Aalto University SoE*.

Siikonen, T., J. Hoffren, and S. Laine (1990). "A Multigrid *LU* Factorization Scheme for the Thin-Layer Navier–Stokes Equations". In: *Proceedings of the 17th ICAS Congress*. ICAS Paper 90-6.10.3. Stockholm, pp. 2023–2034.

Spalart, P.R. and S.R. Allmaras (1994). "A One-equation Turbulence Model for Aerodynamic Flows". In: *La Recherche Aérospatiale* 1.

Spalart, P.R., W-H. Jou, M. Strelets, and S.R. Allmaras (1997). "Comments on the Feasibility of LES for Wings, and on a Hybrid RANS/LES Approach". In: *Advances in DNS/LES* 1, pp. 4–8.

Spalding, D.B. (1991). "Kolmogorov's two-equation model of turbulence". In: *Proceedings of the Royal Society of London* 434.1890, pp. 211–216.

Van Leer, B. (1976). "MUSCL, a new approach to numerical gas dynamics". In: *Computing in Plasma Physics and Astrophysics*.

Wallin, S. and A.V. Johansson (2000). "An explicit algebraic Reynolds stress model for incompressible and compressible turbulent flows". In: *Journal of Fluid Mechanics* 403, pp. 89–132.

Warming, RF and BJ Hyett (1974). "The modified equation approach to the stability and accuracy analysis of finite-difference methods". In: *Journal of computational physics* 14.2, pp. 159–179.

White, F.M. (2006). *Viscous fluid flow*. 3rd ed. New York, USA: McGraw-Hill. 629 pp. ISBN: 007-124493-X.

Wieghardt, K. and W. Tillmann (1944). *Zur turbulenten Reibungsschicht bei Druckanstieg*. Göttingen, Germany: Kaiser Wilhelm-Institut für Strömungsforschung.

Wilcox, D.C. (1988). "Reassessment of the scale-determining equation for advanced turbulence models". In: *AIAA Journal* 26.11, pp. 1299–1310.

— (2008). "Formulation of the $k - \omega$ turbulence model revisited". In: *AIAA Journal* 46.11, pp. 2823–2838.

Appendix: Computed coefficients

Flow solution	Grid level	C_L	C_D	C_M	C_{Dp}
SA Single precision Full friction	1	0.012325	1.0903	0.0068388	0.0061191
	2	0.012301	1.0901	0.0069000	0.0061102
	3	0.012288	1.0893	0.0070741	0.0061072
	4	0.012344	1.0882	0.0072246	0.0061849
	5	0.012750	1.0866	0.0071902	0.0066248
	6	0.019545	1.1337	0.012810	0.0133347
SA Double precision Full friction	2	0.012253	1.09036	0.0068989	0.0060632
	3	0.012252	1.08946	0.0070726	0.0060735
	4	0.012321	1.08834	0.0072272	0.0061609
	5	0.012738	1.08667	0.0071982	0.0066155
	6	0.019540	1.13372	0.0128139	0.0133291
SA Single precision Thin layer	2	0.012309	1.0892	0.0070599	0.0061209
	3	0.012302	1.0882	0.0072564	0.0061266
	4	0.012357	1.0870	0.0074401	0.0061997
	5	0.012764	1.0853	0.0074233	0.0066455
SST Single precision Full friction	3	0.012922	1.0729	0.0095487	0.0066310
	4	0.012954	1.0730	0.0095378	0.0066728
	5	0.013267	1.0728	0.0093473	0.0070399
SST Single precision Thin layer	2	0.012943	1.0728	0.0095331	0.0066623
	3	0.012940	1.0726	0.0095891	0.0066586
	4	0.012976	1.0726	0.0095819	0.0067043
	5	0.013286	1.0723	0.0094085	0.0070665
	6	0.019784	1.1228	0.0145850	0.0134948